INSO

14841-21

1st. Edition

Apr.2013

جمهوری اسلامی ایران
Islamic Republic of Iran
سازمان ملی استاندارد ایران
Iranian National Standardization Organization

استاندارد ملی ایران

۱۴۸۴۱-۲۱

چاپ اول

اردیبهشت ۱۳۹۲

فناوری اطلاعات- اتصال متقابل سامانه‌های باز – مدیریت سامانه‌ها- ترتیب‌سنج فرمان برای مدیریت سامانه‌ها

Information technology— Open Systems Interconnection – Systems Management: Command sequencer for Systems Management

ICS: 35.040

به نام خدا

آشنایی با سازمان ملی استاندارد ایران

مؤسسهٔ استاندارد و تحقیقات صنعتی ایران به موجب بند یک مادهٔ۳ قانون اصلاح قوانین و مقررات مؤسسهٔ استاندارد و تحقیقات صنعتی ایران، مصوب بهمن ماه ۱۳۷۱ تنها مرجع رسمی کشور است که وظیفه تعیین، تدوین و نشر استانداردهای ملی (رسمی) ایران را به عهده دارد.

نام موسسه استاندارد و تحقیقات صنعتی ایران به موجب یکصد و پنجاه و دومین جلسه شورای عالی اداری مورخ ۹۰/۶/۲۹ به سازمان ملی استاندارد ایران تغییر و طی نامه شماره ۲۰۶/۳۵۸۳۸ مورخ ۹۰/۷/۲۴ جهت اجرا ابلاغ شده است .

تدوین استاندارد در حوزه های مختلف در کمیسیون های فنی مرکب از کارشناسان سازمان ، صاحب نظران مراکز و مؤسسات علمی، پژوهشی، تولیدی و اقتصادی آگاه و مرتبط انجام می شود وکوششی همگام با مصالح ملی و با توجه به شرایط تولیدی، فناوری و تجاری است که از مشارکت آگاهانه و منصفانهٔ صاحبان حق و نفع، شامل تولیدکنندگان، مصرف‌کنندگان، صادرکنندگان و وارد کنندگان، مراکز علمی و تخصصی، نهادها، سازمان های دولتی و غیر دولتی حاصل می شود .پیش نویس استانداردهای ملی ایران برای نظرخواهی به مراجع ذی نفع و اعضای کمیسیون های فنی مربوط ارسال می‌شود و پس از دریافت نظرها و پیشنهادها در کمیتهٔ ملی مرتبط با آن رشته طرح و در صورت تصویب به عنوان استاندارد ملی (رسمی) ایران چاپ و منتشر می شود.

پیش نویس استانداردهایی که مؤسسات و سازمان های علاقه مند و ذی صلاح نیز با رعایت ضوابط تعیین شده تهیه می کنند درکمیتهٔ ملی طرح و بررسی و درصورت تصویب، به عنوان استاندارد ملی ایران چاپ و منتشر می شود .بدین ترتیب، استانداردهایی ملی تلقی می شوند که بر اساس مفاد نوشته شده در استاندارد ملی ایران شمارهٔ ۵ تدوین و در کمیتهٔ ملی استاندارد مربوط که سازمان ملی استاندارد ایران تشکیل می‌دهد به تصویب رسیده باشد.

سازمان ملی استاندارد ایران از اعضای اصلی سازمان بین المللی استاندارد (ISO)[1] ،کمیسیون بین المللی الکتروتکنیک (IEC)[2] و سازمان بین المللی اندازه شناسی قانونی (OIML)[3] است و به عنوان تنها رابط[4] کمیسیون کدکس غذایی (CAC)[5] در کشور فعالیت می کند. در تدوین استانداردهای ملی ایران ضمن توجه به شرایط کلی و نیازمندی های خاص کشور ، از آخرین پیشرفت های علمی، فنی و صنعتی جهان و استانداردهای بین‌المللی بهره‌گیری می‌شود .

سازمان ملی استاندارد ایران می تواند با رعایت موازین پیش بینی شده در قانون، برای حمایت از مصرف کنندگان، حفظ سلامت و ایمنی فردی و عمومی، حصول اطمینان از کیفیت محصولات و ملاحظات زیست محیطی و اقتصادی، اجرای بعضی از استانداردهای ملی ایران را برای محصولات تولیدی داخل کشور و/یا اقلام وارداتی، با تصویب شورای عالی استاندارد، اجباری نماید. سازمان می تواند به منظور حفظ بازارهای بین المللی برای محصولات کشور، اجرای استاندارد کالاهای صادراتی و درجه‌بندی آن را اجباری نماید.

همچنین برای اطمینان بخشیدن به استفاده کنندگان از خدمات سازمان ها و مؤسسات فعال در زمینهٔ مشاوره، آموزش، بازرسی، ممیزی و صدورگواهی سیستم های مدیریت کیفیت و مدیریت زیست‌محیطی، آزمایشگاه ها و مراکز کالیبراسیون (واسنجی) وسایل سنجش، سازمان ملی استاندارد ایران این گونه سازمان ها و مؤسسات را بر اساس ضوابط نظام تأیید صلاحیت ایران ارزیابی می کند و در صورت احراز شرایط لازم، گواهینامهٔ تأیید صلاحیت به آن ها اعطا و بر عملکرد آن ها نظارت می کند. ترویج دستگاه بین المللی یکاها، کالیبراسیون (واسنجی) وسایل سنجش، تعیین عیار فلزات گرانبها و انجام تحقیقات کاربردی برای ارتقای سطح استانداردهای ملی ایران از دیگر وظایف این سازمان است.

1- International Organization for Standardization
2 - International Electrotechnical Commission
3- International Organization of Legal Metrology (Organisation Internationale de Metrologie Legale)
4 - Contact point
5 - Codex Alimentarius Commission

کمیسیون فنی تدوین استاندارد

«فناوری اطلاعات – اتصال متقابل سامانه‌های باز –مدیریت سامانه‌ها-ترتیب‌سنج فرمان برای مدیریت سامانه‌ها»

رئیس:	**سمت و/یا نمایندگی**
فولادیان، مجید (فوق لیسانس مهندسی برق مخابرات)	مشاور سازمان فناوری اطلاعات ایران
دبیر:	
میراسکندری، سید محمدرضا (لیسانس مهندسی کامپیوتر نرم افزار)	مدیر کل خدمات ارزش افزوده سازمان فناوری اطلاعات
اعضا : (اسامی به ترتیب حروف الفبا)	
ابریشمی، سعید (دکترای مهندسی کامپیوتر)	استادیار دانشگاه فردوسی مشهد
امیریان، احسان (فوق لیسانس مهندسی کامپیوتر)	مدیر عامل شرکت هوشمندی تجاری تالی
اطمینانی، کبری فوق‌لیسانس مهندسی کامپیوتر نرم‌افزار	نماینده دانشگاه فردوسی مشهد
بختیاری، شیرین (لیسانس مهندسی برق)	کارشناس تدوین استاندارد سازمان فناوری اطلاعات ایران
پایدار، صمد (فوق لیسانس مهندسی کامپیوتر نرم افزار)	نماینده دانشگاه فردوسی مشهد
جمیل پناه، ناصر (فوق لیسانس مدیریت)	کارشناس تدوین استاندارد سازمان فناوری اطلاعات ایران
حدادان، شهره (لیسانس مهندسی کامپیوتر)	نماینده دانشگاه فردوسی مشهد

سعیدی، عذرا (فوق لیسانس مهندسی برق مخابرات)	کارشناس تدوین استاندارد سازمان فناوری اطلاعات ایران
سلطانی حقیقت، الهه (لیسانس مهندسی برق مخابرات)	کارشناس تدوین استاندارد سازمان فناوری اطلاعات ایران
عسگرزاده، مجید (فوق لیسانس مهندسی کامپیوتر)	مدیر پروژه موسسه تحقیقات ارتباطات و فناوری اطلاعات
فرهاد شیخ احمد، لیلا (فوق لیسانس مهندسی کامپیوتر نرم افزار)	کارشناس تدوین استاندارد سازمان فناوری اطلاعات ایران
فیاضی، مهدی (لیسانس مهندسی برق الکترونیک)	کارشناس و مسؤول تدوین استاندارد و امنیت شبکه
قسمتی، سیمین (فوق لیسانس فناوری اطلاعات)	کارشناس تدوین استاندارد سازمان فناوری اطلاعات ایران
قهرمانی، معصومه (فوق لیسانس مهندسی کامپیوتر نرم افزار	کارشناس مرکز آمار و کامپیوتر دانشگاه فردوسی مشهد
قهرمانی، مرضیه (فوق لیسانس مهندسی کامپیوتر نرم افزار	نماینده دانشگاه فردوسی مشهد
معروف، سینا (لیسانس مهندسی کامپیوتر سخت‌افزار)	کارشناس سازمان فناوری اطلاعات ایران
میرزایی رضایی، طیبه (فوق لیسانس فیزیک)	رییس اداره تدوین استانداردها و نظارت بر امنیت سرویس‌ها سازمان فناوری اطلاعات ایران

فهرست مندرجات

پیش‌گفتار

استاندارد فناوری اطلاعات- اتصال متقابل سامانه‌های باز – مدیریت سامانه‌ها- قسمت ۲۱: ترتیب‌سنج فرمان برای مدیریت سامانه‌ها» که پیش‌نویس آن در کمیسیون‌های مربوط به‌وسیله سازمان فناوری اطلاعات ایران تهیه و تدوین شده و در دویست و پنجاه و هشتمین اجلاس کمیته‌ی ملی استاندارد رایانه و فرآوری داده‌ها مورخ ۱۳۹۱/۱۱/۱۴ مورد تصویب قرارگرفته است، اینک به استناد بند یک ماده‌ی ۳ قانون اصلاح قوانین و مقررات موسسه استاندارد و تحقیقات صنعتی ایران، مصوب بهمن‌ماه ۱۳۷۱، به‌عنوان استاندارد ملی ایران منتشر می‌شود.

برای حفظ همگامی و هماهنگی با تحولات و پیشرفت‌های ملی و جهانی در زمینه‌ی صنایع، علوم و خدمات، استانداردهای ملی ایران در مواقع لزوم تجدید نظر خواهد شد و هر پیشنهاد که برای اصلاح و تکمیل این استانداردها ارائه شود، هنگام تجدید نظر در کمیسیون فنی مربوط مورد توجه قرار خواهد گرفت. بنابراین، باید همواره از آخرین تجدید نظر استانداردهای ملی استفاده کرد.

منبع و مأخذی که برای تدوین این استاندارد مورد استفاده قرار گرفته به شرح زیر است:

ISO/IEC 10164-21:1998, *Information technology— Open Systems Interconnection – Systems Management: Command sequencer for Systems Management*

مقدمه

استاندارد بین‌المللی ISO/IEC 10164-21 به‌وسیله کمیته فنی مشترک ISO/IEC JTC 1، فناوری اطلاعات، زیرکمیته SC33، خدمات برنامه کاربردی توزیع‌شده، با همکاری ITU-T آماده شده است. همان متن به‌عنوان توصیه نامه ITU-T X.753 منتشر شده است.

ISO/IEC 10164 تحت عنوان کلی فناوری اطلاعات- اتصال متقابل سامانه‌های باز- مدیریت سامانه، شامل قسمت‌های زیر است:

- قسمت ۱: تابع مدیریت شی
- قسمت ۲: تابع مدیریت وضعیت
- قسمت ۳: صفاتی برای نشان دادن رابطه‌ها
- قسمت ۴: تابع گزارش اخطار
- قسمت ۵: تابع مدیریت گزارش روبداد
- قسمت ۶: تابع گزارش لاگ[1]
- قسمت ۷: تابع گزارش اخطار امنیتی
- قسمت ۸: تابع پیگیری ممیزی امنیتی
- قسمت ۹: اشیاء و صفات برای کنترل دسترسی
- قسمت ۱۰: تابع سنجش کاربرد برای اهداف حسابرسی
- قسمت ۱۱: صفات و اشیاء سنجی
- قسمت ۱۲: تابع مدیریت آزمون
- قسمت ۱۳: تابع خلاصه‌سازی
- قسمت ۱۴: رده‌های آزمون اعتماد و تشخیص
- قسمت ۱۵: تابع زمان‌بندی
- قسمت ۱۶: تابع مدیریت دانش مدیریتی
- قسمت ۱۷: تابع تغییر
- قسمت ۱۸: تابع مدیریت نرم‌افزار
- قسمت ۱۹: توابع مدیریت دامنه مدیریت و سیاست مدیریتی
- قسمت ۲۰: تابع مدیریت زمان
- قسمت ۲۱: ترتیب‌سنج فرمان برای مدیریت سامانه‌ها
- قسمت ۲۲: تابع نظارت بر زمان پاسخ

1- Log

پیوست‌های الف، ب، ت و ج تا ح یک قسمت کامل از این قسمت از استاندارد ISO/IEC 10164 را شکل می‌دهند.
پیوست‌های پ و ث فقط اطلاعاتی هستند.

فناوری اطلاعات – ارتباط متقابل سامانه‌های باز – مدیریت سامانه‌ها- قسمت ۲۱: ترتیب‌سنج فرمان برای مدیریت سامانه‌ها

۱ هدف و دامنه کاربرد

هدف از تدوین این استاندارد، تعریف تابع مدیریت سامانه‌ها است که همان‌طورکه در استاندارد ISO/IEC 7498-4 یا توصیه‌نامه CCITT Rec. X.700[1] تعریف شده است، می‌تواند به‌وسیله یک فرآیند برنامه کاربردی، برای تعامل به‌منظور مدیریت سامانه‌ها در یک محیط مدیریتی متمرکز یا غیرمتمرکز، استفاده شود. این استاندارد، ترتیب‌سنج فرمان را که شامل تعاریف عمومی، خدمات و واحدهای عملکردی است، تعریف می‌کند. این تابع در لایه کاربرد IUT-T Rec. X.200 | ISO/IEC 7498-1 قرار گرفته است و مطابق مدل ارائه شده به‌وسیله استاندارد ISO 9545 تعریف می‌شود. نقش توابع مدیریت سامانه‌ها به‌وسیله ITU Rec. X.701 | ISO/IEC 10040 توصیف می‌شود.

این استاندارد برای موارد زیر کاربرد دارد:

- نیازمندی‌های کاربر برای ترتیب‌سنج فرمان را برقرار می‌کند؛
- مدل‌هایی ایجاد می‌کندکه خدمات ارائه شده به‌وسیله تابع را به نیازهای کاربر مربوط می‌سازد؛
- خدمات ارائه شده به‌وسیله تابع را تعریف می‌کند؛
- پروتکل موردنیاز به‌منظور ارائه خدمات را مشخص می‌کند؛
- ارتباط بین خدمات و عملیات SMI و اعلان‌ها[2] را تعریف می‌کند؛
- ارتباطات با دیگر توابع مدیریت سامانه‌ها را تعریف می‌کند؛
- نیازمندی‌های انطباق را تعریف می‌کند؛
- یک زبان اسکریپت‌نویسی[3] برای استفاده در محیط ترتیب‌سنج فرمان تعریف می‌کند.

این استاندارد برای موارد زیر کاربرد ندارد:

- تعریف ماهیت هر پیاده‌سازی که قصد ارائه ترتیب‌سنج فرمان را دارد؛
- مشخص کردن شیوه انجام مدیریت با استفاده از ترتیب‌سنج فرمان؛
- تعریف ماهیت هر دستورالعملی که منجر به استفاده از ترتیب‌سنج فرمان می‌شود ؛
- مشخص کردن خدمات مورد نیاز برای استقرار، انتشار طبیعی، غیرطبیعی مشارکت‌های مدیریت.

1- Comite Consultatif International Telegraphique et Telephonique
2 -Notification
3- Scripting

۲ مراجع الزامی

مدارک الزامی زیر حاوی مقرراتی است که در متن این استاندارد ملی ایران به آن‌ها ارجاع داده شده است . بدین ترتیب آن مقررات جزئی از این استاندارد ملی ایران محسوب می‌شود.

در صورتی که به مدرکی با ذکر تاریخ انتشار ارجاع داده شده باشد، اصلاحیه‌ها و تجدیدنظرهای بعدی آن مورد نظر این استاندارد ملی ایران نیست. در مورد مدارکی که بدون ذکر تاریخ انتشار به آن‌ها ارجاع داده شده‌است، همواره آخرین تجدیدنظر و اصلاحیه‌های بعدی آنها مورد نظر است.

استفاده از مراجع زیر برای این استاندارد الزامی است: [۱]

2-1 IUT-T Recommendation X.200 (1994) | ISO/IEC 7498-1:1994, Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model.

2-2 IUT-T Recommendation X.210 (1993) | ISO/IEC 10731:1994, Information technology – Open Systems Interconnection – Basic Reference Model: Conventions for the definition of OSI services.

2-3 CCITT Recommendation X.701 (1992) | ISO/IEC 10040:1992, Information technology – Open Systems Interconnection – Systems management overview.

2-4 IUT-T Recommendation X.710 (1997) | ISO/IEC 5995:1998, Information technology – Open Systems Interconnection – Common management information service.

2-5 IUT-T Recommendation X.711 (1997) | ISO/IEC 9596-1:1998, Information technology – Open Systems Interconnection – Common management information protocol: Specification.

2-6 CCITT Recommendation X.721 (1992) | ISO/IEC 10165-2:1992, Information technology – Open Systems Interconnection – Structure of management information: Definition of management information.

2-7 CCITT Recommendation X.722 (1992) | ISO/IEC 10165-4:1992, Information technology – Open Systems Interconnection – Structure of management information: Guidelines for the definition of managed objects.

2-8 IUT-T Recommendation X.724 (1996) | ISO/IEC 10165-6:1997, Information technology – Open Systems Interconnection – Structure of management information: Requirements and guidelines for implementation conformance statement performas associated with OSI management.

2-9 IUT-T Recommendation X.725 (1995) | ISO/IEC 10165-7:1996, Information technology – Open Systems Interconnection – Structure of management information: General relationship model.

2-10 CCITT Recommendation X.730 (1992) | ISO/IEC 10164-1:1993, Information technology – Open Systems Interconnection – System management: Object management function.

2-11 CCITT Recommendation X.731 (1992) | ISO/IEC 10164-2:1993, Information technology – Open Systems Interconnection – System management: State management function.

2-12 CCITT Recommendation X.733 (1992) | ISO/IEC 10164-4:1993, Information technology – Open Systems Interconnection – System management: Alarm reporting function.

2-13 CCITT Recommendation X.734 (1992) | ISO/IEC 10164-5:1993, Information technology – Open Systems Interconnection – System management: Event report management function.

---

[۱] - مراجع الزامی بندهای ۱-۲ الی ۱۷-۲ مربوط به توصیه‌نامه‌ها | استانداردهای بین‌المللی همسان و مراجع الزامی بندهای ۱۸-۲ الی ۲۵-۲ مربوط به توصیه‌نامه‌ها | استانداردهای بین‌المللی معادل از نظر محتوای فنی هستند.

2-14 CCITT Recommendation X.735 (1992) | ISO/IEC 10164-6:1993, Information technology – Open Systems Interconnection – System management: Log control function.
2-15 IUT-T Recommendation X.739 (1993) | ISO/IEC 10164-11:1994, Information technology – Open Systems Interconnection – System management: Metric objects and attributes.
2-16 IUT-T Recommendation X.741 (1995) | ISO/IEC 10164-9:1995, Information technology – Open Systems Interconnection – System management: Objects and attributes for access control.
2-17 IUT-T Recommendation X.746 (1995) | ISO/IEC 10164-15:1995, Information technology – Open Systems Interconnection – System management: Scheduling function.
2-18 CCITT Recommendation X.209 (1988), Specification of basic encoding rules for Abstract Syntax Notation One (ASN.1).
2-19 ISO/IEC 8825:1990, Information technology – Open Systems Interconnection – Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1).
2-20 ITU-T Recommendation X.291 (1995), OSI conformance testing methodology and framework for protocol Recommendation for IUT-T applications – Abstract test suite specification.
2-21 ISO/IEC 9646-2:1994, Information technology – Open Systems Interconnection – Conformance testing methodology and framework – Part 2: Abstract Test Suite specification.
2-22 ITU-T Recommendation X.296 (1995), OSI conformance testing methodology and framework for protocol Recommendations for IUT-T applications – Implementation conformance statement.
2-23 ISO/IEC 9646-7:1995, Information technology – Open Systems Interconnection – Conformance testing methodology and framework – Part 7: Implementation Conformance Statements.
2-24 CCITT Recommendation X.700 (1992), Management framework for Open Systems Interconnection (OSI) for CCITT Applications.
2-25 ISO/IEC 7498-4:1989, Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 4: Management framework.

۳ اصطلاحات و تعاریف

در این استاندارد اصطلاحات و تعاریف زیر به‌کار می‌روند.

۳-۱ تعاریف مدل مرجع پایه

این استاندارد از اصطلاحات زیر که در ITU-T X.200 | ISO/IEC7498-1 تعریف شده‌اند، استفاده می‌کند.

۳-۱-۱

سامانه باز[1]؛

۳-۱-۲

مدیریت سامانه‌ها[2].

۳-۲ تعاریف قرارداد خدمت

این استاندارد از اصطلاح زیر که در ITU-T X.210 | ISO/IEC10731 تعریف شده است، استفاده می‌کند.

---

1 - Open system
2 - Systems management

۳-۲-۱

عمل پایه‌ای[1].

۳-۳ تعاریف چارچوب مدیریت

این استاندارد از اصطلاحات زیر که در CCITT X.700|ISO/IEC7498-4 تعریف شده‌اند، استفاده می‌کند.

۳-۳-۱

اطلاعات مدیریتی[2]؛

۳-۳-۲

شیء مدیریت شده[3].

۳-۴ تعاریف مرور کلی مدیریت سامانه‌ها

این استاندارد از اصطلاحات زیر که در CCITTX.701|ISO/IEC10040 تعریف شده‌اند، استفاده می‌کند.

۳-۴-۱

نقش عامل[4]؛

۳-۴-۲

شیء پشتیبانی مدیریت[5]؛

۳-۴-۳

کلاس شیء مدیریت شده[6]؛

۳-۴-۴

نقش مدیر[7]؛

۳-۴-۵

اعلان[8]؛

۳-۴-۶

واحد کارکرد مدیریت سامانه‌ها[9]؛

---

1 -Primitive
2- Management information
3- Managed object
4 -Agent role
5 -Management support object
6 -Managed object class
7 -Manager role
8 -Notification
9 -Systems management function unit

۳-۴-۷

عملیات مدیریت سامانه[1].

۳-۵ تعاریف خدمت اطلاعات مدیریتی مشترک

این استاندارد از اصطلاحات زیر که در ITU-T X.701|ISO/IEC9595 تعریف شده‌اند، استفاده می‌کند.

۳-۵-۱

صفت[2]؛

۳-۵-۲

خدمات اطلاعات مدیریت مشترک[3].

۳-۶ تعاریف اضافی

اصطلاحات زیر در این استاندارد تعریف شده‌اند.

۳-۶-۱ ترتیب‌سنج فرمان[4]

یک شیء پشتیبان مدیریت که نشان‌دهنده منبعی در نقش یک مدیر به‌عنوان یک مقصد اعلان است و به‌عنوان یک آغازگر عملیات تعیین‌شده به‌وسیله اسکریپت‌های راه‌اندازی آن، با توانایی محول کردن فعالیت‌های مدیریتی، عمل می‌کند.

۳-۶-۲ اسکریپت راه‌اندازی[5]

یک شیء مدیریت شده که نشان‌دهنده دستورالعمل‌هایی است که باید به‌وسیله یک ترتیب‌سنج فرمان اجرا شوند.

۳-۶-۳ نخ[6]

یک شیء مدیریت شده که نشان‌دهنده اجرای یک اسکریپت راه‌اندازی است. نتایج اجرا یا خطاهای حاصل از اجراهای اسکریپت راه‌اندازی به‌وسیله نخ برگردانده می‌شوند.

۳-۶-۴ نخ تعلیق‌پذیر[7]

نخ تعلیق‌پذیر از کلاس شیء مدیریت‌شده نخ، مشتق شده است. این نخ‌ها به‌وسیله لایه‌های راه‌انداز غیرهمزمان تولید می‌شوند. آن‌ها می‌توانند به‌وسیله اقدام تعلیق ارسال‌شده به آن‌ها معلق شوند و به‌وسیله یک اقدام ازسرگیری ارسال‌شده به آنها، ازسر گرفته شوند.

1- System management operation
2 -Attribute
3 -Common management information services
4 -Command sequencer
5 -Launch script
6 -thread
7 -Suspendable thread

۳-۶-۵ لت راه‌اندازی[1]

شیء پشتیبان مدیریت که ممکن است یک ماشه‌چکانی[2] برای آغاز اجرای یک اسکریپت راه‌اندازی، به آن ارسال شود. لت راه‌اندازی به‌عنوان یک شیء مدیریت شده مقدار اولیه (IVMO)[3] برای یک نخ عمل می‌کند.

۳-۶-۶ لت راه‌اندازی غیرهمزمان[4]

لت راه‌انداز غیرهمزمان از لت راه‌اندازی مشتق می‌شود. این لت، اعلان نتیجه ماشه‌چکانی را بدون منتظر شدن برای نتایج اجرای اسکریپت‌های راه‌اندازی، برمی‌گرداند. نتایج اجرا یا خطاهای اجراهای اسکریپت راه‌اندازی به‌طور مستقیم از طریق نخ اعلام می‌شوند.

۳-۶-۷ لت راه‌اندازی همزمان[5]

لت راه‌انداز همزمان از لت راه‌اندازی مشتق می‌شود. این لت، پس از آنکه نخ‌ها اجرای خود را به پایان رساندند، تمام نتایج اجرا و خطاها را از نخ‌ها دریافت کرده و اعلان نتیجه ماشه‌چکانی یا اخطار خطای پردازش را برمی‌گرداند.

۳-۶-۸ فعال‌ساز ماشه‌چکانی[6]

آغازگر اجرای اسکریپت از طریق وادار کردن یک لت راه‌اندازی برای تولید یک یا چند نخ است. این فعال‌ساز فرمان را به شکل یک زمان‌بند، عملیات، اعلان‌ها یا اقدام محلی به یک لت راه‌اندازی هدایت می‌کند.

۳-۶-۹ فرمان[7]

دستورالعملی برای یک فعالیت مدیریت که در سامانه عامل مطابق با محتوای یک اسکریپت راه‌اندازی اجرا می‌شود. فرمان با یک زبان اسکریپت‌نویسی توصیف می‌شود. در حال حاضر، زبان اسکریپت‌نویسی مدیریت سامانه در پیوست پ توصیف شده است.

۴ کوته‌نوشت‌ها

در این استاندارد ، اصطلاحات کوته‌نوشت زیر به‌کار می‌روند:

ANS1	Abstract Syntax Notation One	نشان گذاری نحو انتزاعی یک
CMIS	Common Management Information Service	خدمت اطلاعات مدیریت مشترک
CS	Common Sequencer	ترتیب سنج فرمان
IVMO	Initial Value Managed Object	شیء مدیریت شده مقدار اولیه

1- Launch pad
2- Trigger
3- Initial Value Managed Object
4- Asynchronous launch pad
5- Sunchronous launch pad
6- Trigger activator
7- Command

OSI	Open System Interaction	اتصال متقابل سیستم‌های باز
LP	Launch Pad	لت راه اندازی
SMSL	Systems Management Scripting Language	زبان اسکریپ نویسی مدیریت سامانه ها

۵ قراردادها[1]

این مشخصه، خدماتی را برای ترتیب‌سنج فرمان تعریف می‌کند که قراردادهای توصیفی زیر را که در ITU-T Rec. X.210 | ISO/IEC10731 تعریف شده‌اند، دنبال می‌کند. در بند ۹، تعریف هر خدمت شامل جدولی است که پارامترهای خدمت را فهرست می‌کند. برای یک عمل پایه‌ای خدمت داده شده، حضور هر پارامتر با یکی از مقادیر زیر توصیف می‌شود:

M	پارامتر اجباری است.
(=)	مقدار پارامتر با مقدار پارامتر در ستون سمت چپ برابر است.
U	استفاده از پارامتر، یک انتخاب کاربر خدمت است.
-	پارامتر در تعامل توصیف‌شده به‌وسیله عمل پایه‌ای مورد نظر وجود ندارد.
C	پارامتر شرطی است. شرایط به‌وسیله آزمونی که این پارامتر را توصیف می‌کند، تعریف می‌شوند.
P	پارامتر تحت محدودیت‌های اعمال شده به‌وسیله ITU-T Rec. X.710 | ISO/IEC 9595قرار دارد.

یادآوری- پارامترهایی که در جداول خدمت این مشخصه با **P** علامت خورده‌اند، بدون تغییر معانی یا نحو پارامترها، به‌طور مستقیم به پارامترهای متناظر از عمل پایه‌ای خدمت **CMIS** نگاشت می‌شوند.

۶ الزامات

الزاماتی که باید برآورده شوند عبارتند از:

- الزامات کاربر:
  - اجازه محول کردن فعالیت‌های مدیریتی.
  - کاهش میزان ارتباطاتی که باید میان مدیر و عامل‌ها رخ دهد.
  - اجازه به سامانه‌های مدیر نماینده، برای کار روی سامانه‌های عامل ، حتی زمانی‌که ارتباطات میان مدیر و سامانه‌های عامل مختل شده یا ممکن نیست.
  - مهیاکردن کنترل انعطاف‌پذیر فعالیت‌های مدیریت.
  - مهیاکردن یک زبان اسکریپت‌نویسی که می‌تواند رویه‌هایی که عملیات مدیریت را اجرا می‌کنند، توصیف کند.
  - اجازه دادن به سامانه‌های نماینده برای اجرای به‌ترتیب عملیات CMIS.
- الزامات عملیاتی:

1- Conventions

- اجرای از قبل زمانبندی‌شده یا به تعویق افتاده یک عمل مدیریت سامانه‌ها.
- قابلیت‌هایی برای تغییر درخواست اجرای از پیش زمانبندی‌شده یا به تعویق افتاده.
- قابلیت‌هایی برای شروع، تعلیق، ازسرگیری و پایان عملیات مدیریت سامانه‌ها براساس اقدامات مدیریت زمان یا وقوع رویدادها.
- قابلیت‌هایی برای گزارش‌دهی و ثبت خروجی اجرای از پیش زمان‌بندی شده یا به تعویق افتاده.
- توانایی ارسال اعلانات، هنگامی‌که تغییر وضعیت رخ می‌دهد.

۷ مدل

۷-۱ توصیف مدل

مدل نحوه اجرای ماشه‌چکانی‌شده، ازپیش زمان‌بندی‌شده یا به‌تعویق‌افتاده‌ی عملیات مدیریت سامانه‌ای را که می‌تواند به‌وسیله ترتیب‌سنج فرمان انجام شود، توصیف می‌کند. مدل مؤلفه‌های مفهومی، رابطه میان این مؤلفه‌ها، توصیفی از وضعیت‌ها و انتقال وضعیت‌های ممکن را تشریح می‌کند.

شکل ۱ توصیف نمادینی از قابلیت ترتیب‌سنج فرمان یک سامانه است.

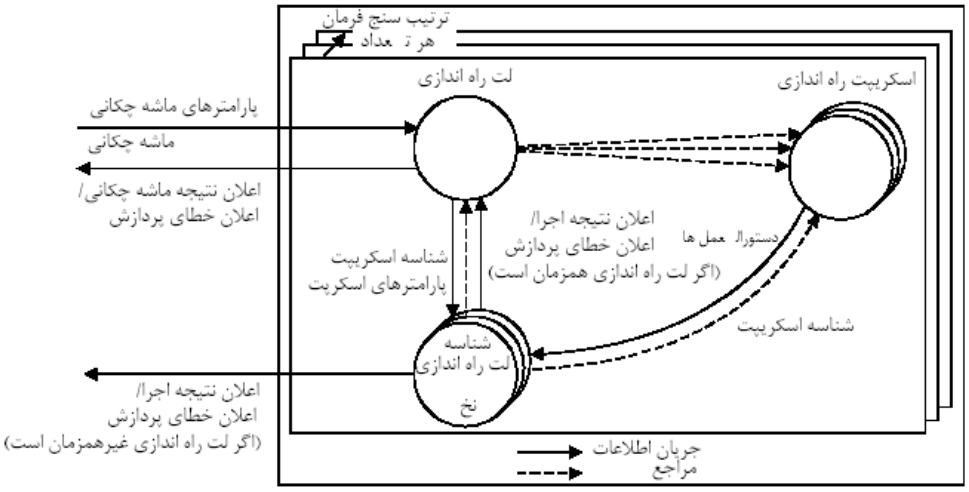
عملکرد یک ترتیب‌سنج فرمان به‌وسیله اسکریپت راه‌اندازی، نخ و اشیاء لت راه‌اندازی مدل می‌شود. مدل مذکور یک انتزاع OSI از اجرای عملیات ازپیش زمانبندی‌شده یا به‌تعویق‌افتاده در سامانه‌های باز است. یک ترتیب‌سنج فرمان می‌تواند شامل هر تعداد لت راه‌اندازی باشد که ترتیب‌سنج فرمان برای آنها به‌عنوان یک ارائه‌دهنده‌ی خدمات عمل می‌کند. هر لت راه‌اندازی ممکن است در هر لحظه یک اسکریپت راه‌اندازی، یا چندین اسکریپت راه‌اندازی را اجرا کند. با دریافت یک ماشه‌چکانی از یک فعال‌ساز ماشه‌چکانی، لت راه‌اندازی، اجرای یک اسکریپت راه‌اندازی را آغاز می‌کند. علاوه بر مؤلفه شناسه ماشه‌چکانی، ماشه‌چکانی ممکن است یک نام اسکریپت راه‌اندازی (شناسه اسکریپت) و آرگومان‌های ورودی به اسکریپت را به‌عنوان پارامترهایی درون مؤلفه فهرست پارامتر اجرا، مشخص کند.

دو نوع لت راه‌اندازی وجود دارد، لت راه‌اندازی غیرهمزمان و لت راه‌اندازی همزمان. یک لت راه‌اندازی غیرهمزمان، اعلان نتیجه ماشه‌چکانی را بدون انتظار برای نتایج اجرای اسکریپت‌های راه‌اندازی برمی‌گرداند. نتایج اجرا یا خطاهای اجراهای اسکریپت راه‌اندازی به‌طور مستقیم از طریق نخ اعلان می‌شوند. از طرف دیگر، یک لت راه‌اندازی همزمان، اعلان نتیجه ماشه‌چکانی یا اخطار خطای پردازش را پس از اینکه همه نتایج اجرا و خطاها را از نخ‌ها، بعد از تکمیل اجرای آنها دریافت کرد، بازمی‌گرداند.

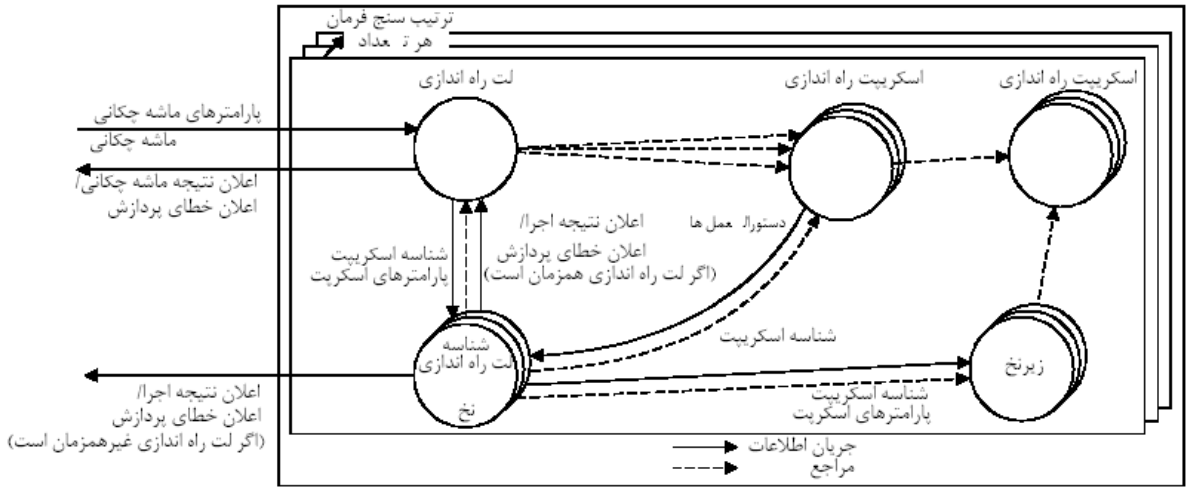
یک نمونه اسکریپت راه‌اندازی ممکن است به هر تعداد دستورالعمل‌های یکتا داشته باشد. مؤلفه فهرست پارامتر اجرای ماشه‌چکانی، فهرستی از اسکریپت‌ها است که باید اجرا شوند (و به‌وسیله شناسه اسکریپت خود شناسایی می‌شوند) و برای اجرای آن‌ها به پارامترهای ورودی مربوطه نیاز است. ممکن است یک فهرست پارامتر اجرای پیش‌فرض، برای لت راه‌اندازی مشخص شود تا بتواند در مواردی که ماشه‌چکانی‌ای دریافت می‌کند که پارامترهای آن مشخص نشده‌اند، اجرا شود. اگر لت راه‌اندازی برای اجرای یک فهرست پارامتر اجرای پیش‌فرض پیکربندی نشده باشد و مؤلفه فهرست پارامتر اجرا به‌وسیله ماشه‌چکانی تهیه نشده باشد و اگر لت راه‌اندازی یک ماشه‌چکانی دریافت کند که سعی در فعال‌سازی آن دارد، یک کد خطای بدون اسکریپت در فیلد کد خطای اعلان نتیجه

ماشه‌چکانی برگردانده می‌شود. لت راه‌اندازی یک صفت فهرست اسکریپت در اختیار دارد که می‌تواند برای شناسایی اسکریپت‌هایی که به‌وسیله آن اجرا می‌شوند، پیکربندی شود. اگر یک مؤلفه فهرست پارامتر اجرا در ماشه‌چکانی موجود باشد، لت راه‌اندازی وجود شناسه اسکریپت از مؤلفه پارامتر اجرا در صفت فهرست اسکریپت موجود آن را تأیید می‌کند . فقط نمونه اسکریپت‌های مشخص‌شده به‌وسیله شناسه اسکریپت که در صفت فهرست اسکریپت موجود حضور دارند، اجرا می‌شوند. اگر هیچ‌یک از شناسه‌های اسکریپت در فهرست اسکریپت موجود حضور نداشته باشند، لت راه‌اندازی یک کد خطای اسکریپت ردشده را در نتیجه ماشه‌چکانی برمی‌گرداند و اجرای اسکریپت انجام نمی‌شود.

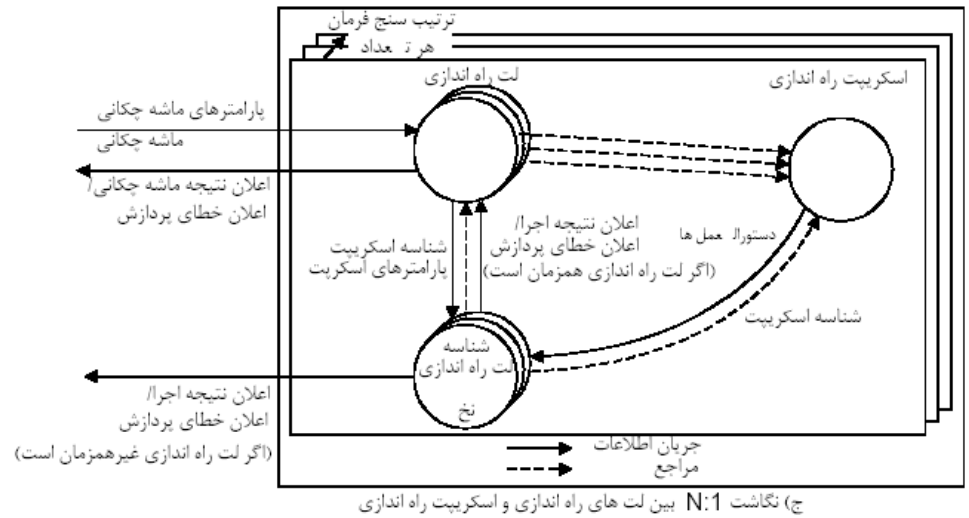
کلاس‌های شیء زبان اسکریپت نویسی ویژه‌سازی‌شده، از کلاس شیء اسکریپت راه‌اندازی مشتق می‌شوند. از این‌رو این دستورالعمل‌ها ممکن است به‌عنوان نمونه‌های اسکریپت ویژه‌سازی‌شده مشخص شوند. مجموعه‌های متعددی از دستورالعمل‌های اسکریپت راه‌اندازی ممکن است به‌صورت ترتیبی یا موازی، مطابق با نوع داده پارامتر اجرا، به‌وسیله نخ‌ها اجرا شوند. ممکن است سطوح تو در توی متعددی از زیرنخ سازی به‌منظور اجرای نمونه‌های اسکریپت لازم باشد.



الف) مورد نمونه



ب) استفاده از زیرنخ



ج) نگاشت N:1 بین لت های راه اندازی و اسکریپت راه اندازی

شکل ۱- مدل ترتیب‌سنج فرمان

برای آغاز رفتار اجرای نمونه‌های اسکریپت راه‌اندازی، باید یک ماشه‌چکانی به نمونه شیء لت راه‌اندازی ارسال شود. ماشه‌چکانی‌های بدون پارامتر ممکن است در مواردی که لت راه‌اندازی یک فهرست پارامتر اجرای پیش فرض دارد، آن لت را فعال کنند.

لت راه‌اندازی به‌عنوان یک IVMO برای نخ عمل می‌کند و فهرست پارامتر اجرا را برای صفت پارامترهای اجرای نخ تهیه می‌کند. فهرست پارامتر اجرا می‌تواند پارامتر اجرای واحد، دنباله‌ای از پارامترهای اجرا یا مجموعه‌ای از پارامترهای اجرا باشد. پارامتر اجرا یک دنباله از شناسه‌های اسکریپت و پارامترهای اسکریپت است. شناسه اسکریپت، نام نمونه شیء مدیریت‌شده از نمونه شیء اسکریپت‌نویسی که باید اجرا شود را شناسایی می‌کند و پارامترهای اسکریپت مقادیر پارامتری را که به‌عنوان ورودی‌های نمونه شیء اسکریپت‌نویسی مورد نیاز هستند، عرضه می‌کند. اگر دنباله‌ای از پارامترهای اجرا مشخص شود، لت راه‌اندازی یک نخ را برای اجرای نمونه‌های اسکریپت ایجاد می‌کند و شناسه اسکریپت و پارامترهای اسکریپت (درصورت نیاز) را از پارامترهای اجرا به نخ ارائه می‌کند. با تکمیل نخ، این کار برای بقیه شناسه‌های اسکریپت درون فهرست، به ترتیب تکرار می‌شود. اگر مجموعه‌ای از پارامترهای اجرا مشخص شوند، لت راه‌اندازی مجموعه‌ای از شناسه‌های اسکریپت و پارامترهای اسکریپت (درصورت نیاز) را برای نخ‌ها آماده می‌کند و نمونه اسکریپت‌های مربوطه به‌صورت موازی اجرا می‌شوند. مفاهیم ارسال پارامتر بین لت راه‌اندازی و نخ‌ها به سازوکار ارجاع پارامتر پشتیبانی‌شده به‌وسیله زبان اسکریپت‌نویسی که اسکریپت به‌وسیله آن نوشته شده است، بستگی دارد.

یک نخ به اجرای یک نمونه اسکریپت اختصاص می‌یابد. این نخ درصورت لزوم ممکن است نخ‌های دیگری ایجاد کند. این امر زمانی ممکن است رخ دهد که یک نمونه اسکریپت، نمونه‌های اسکریپت دیگری را فراخوانی کند. زمانی که این اتفاق رخ می‌دهد، نخی که اسکریپت فراخوانی‌کننده را اجرا می‌کند، یک زیرنخ ایجاد کرده و شناسه و پارامترهای اسکریپت (درصورت نیاز) مربوط به اسکریپت فراخوانده‌شده را به زیرنخ ارسال می‌کند. مفاهیم ارسال پارامتر میان نخ‌ها و زیرنخ‌ها به سازوکار ارسال پارامتر پشتیبانی‌شده به‌وسیله زبان اسکریپت نویسی که اسکریپت با استفاده از آن نوشته شده است، بستگی دارد.

لایه‌های راه‌اندازی غیرهمزمان باید نخ‌های تعلیق‌پذیر ایجاد کنند. نخ تعلیق‌پذیر می‌تواند از طریق اقدامات تعلیق و ازسرگیری، به‌ترتیب معلق و ازسر گرفته شود. نخ‌های یکتای ایجادشده به‌وسیله یک رویه راه‌اندازی همزمان نمی‌توانند تعلیق یا ازسرگیری شوند. تمام نخ‌های مربوط به اجرای یک اسکریپت در هر دو مورد می‌توانند معلق و از سر گرفته شوند.

یک نخ، بعد از اینکه تمام زیرنخ‌هایش با موفقیت تکمیل شدند، یا یک خطا گزارش دادند، کامل می‌شود. به محض وقوع این امر، اجرای یک اسکریپت راه‌اندازی تکمیل می‌شود؛ سپس لت راه‌اندازی مربوطه یک وضعیت غیرفعال (بی‌کار) برمی‌گرداند. شیئی که یک نخ را ایجاد کرده است، دربرگیرنده آن است. یک لایه راه‌اندازی یا یک نخ دیگر می‌توانند در برگیرنده نخ باشند.

چندین لت راه‌اندازی ممکن است به یک اسکریپت راه‌اندازی ویژه ارجاع کنند. نخ‌های متعدد ممکن است به همان اسکریپت راه‌اندازی ارجاع کنند. وجود یک اسکریپت، از هر ارجاعی به‌وسیله لایه‌های راه‌اندازی یا نخ‌ها به آن مستقل است. اسکریپت‌های راه‌اندازی در کلاس‌های اسکریپت ویژه‌سازی‌شده‌ای که از کلاس شیء اسکریپت

راه‌اندازی مشتق شده‌اند، تعریف می‌شوند. مفاهیم و نحو این اسکریپت‌ها در تعریف زبان اسکریپت‌نویسی که اسکریپت‌ها به‌وسیله آن نوشته شده‌اند، مشخص می‌شود. همچنین تعریف زبان اسکریپت‌نویسی، مجموعه‌ای از توابع اسکریپت‌نویسی پایه را که برای تأمین توانایی کنترل و پردازش اسکریپت‌های راه‌اندازی ضروری هستند، مشخص می‌کند.

کلاس شیء مدیریت‌شده اسکریپت رشته‌ای عمومی باید برای نوشتن اسکریپت‌هایی که به شکل یک رشته عمومی ارائه شده‌اند، استفاده شود. صفت نام زبان اسکریپت نشان‌دهنده نام آن زبان اسکریپت‌نویسی است و صفت محتوای اسکریپت، نشان‌دهنده اسکریپتی است که در این زبان اسکریپت‌نویسی در قالب یک رشته عمومی نوشته شده است. پیوست‌های ج و چ یک زبان اسکریپت‌نویسی ویژه‌سازی‌شده، به نام زبان اسکریپت‌نویسی مدیریت سامانه (SMSL)[1] را تعریف می‌کند که بهتر است اسکریپت‌هایی با آن نوشته شوند که به شکل یک رشته عمومی نشان داده می‌شوند. امکان تعریف کلاس‌های دیگری از اسکریپت‌ها وجود دارد. پیوست ت، cmisScript را معرفی می‌کند که یک زبان اسکریپت نویسی در قالب اشیاء مدیریت شده‌ای است که می‌توانند برای نوشتن اسکریپت‌ها در محیط CMIS استفاده شوند.

۷-۲ فرآیند ماشه‌چکانی و گزارش نتایج

فعال کننده‌های ماشه‌چکانی که به لت راه‌انداز ارسال می‌شوند، می‌توانند به اشکال مختلفی مانند زمان‌بندها، عملیات، اعلان‌ها و اقدام محلی باشند. هنگامی‌که لت راه‌اندازی یک ماشه‌چکانی را دریافت می‌کند، یک یا چند نخ برای اجرای اسکریپت ایجاد می‌کند. بعد از اینکه تمام نخ‌های مرتبط با یک ماشه‌چکانی تولید شدند، لت راه‌انداز غیرهمزمان، یک اعلان نتیجه ماشه‌چکانی صادر می‌کند که حاوی مجموعه‌ای از شناسه نخ و شناسه اسکریپت است. نتایج اجرای اسکریپت به‌عنوان اعلان نتیجه اجرا به‌وسیله نخ‌ها و در مورد لت راه‌اندازی غیرهمزمان به‌طور مستقیم به مدیر، منتشر می‌شوند. پس از این‌که همه نخ‌های مربوط به ماشه‌چکانی کامل شدند، لت راه‌اندازی همزمان تمام نتایج اجرا یا خطاهای نخ‌ها را همزمان می‌کند و یک اعلان نتیجه ماشه‌چکانی که حاوی مجموعه‌ای از شناسه نخ، شناسه اسکریپت و نتایج اجرا یا خطاها است را برای مدیر صادر می‌کند.

صفت نوع نتیجه اجرای اسکریپت، نوع نتیجه مورد انتظار از اجرای اسکریپت را تعیین می‌کند و باید مرتبط با صفت نوع نتیجه اجرای مربوط به نتیجه اجرا باشد. هنگامی‌که اجرا موفقیت‌آمیز باشد، فیلد errorCode از نتیجه اجرا به no error code تنظیم شده و در غیر این‌صورت به کد خطای مناسب تنظیم می‌شود.

یک نخ درحال اجرا ممکن است به‌صورت خود به خود، پس از اتمام اجرا یا در شرایط غیرطبیعی متوقف شود . در مورد دوم، نخ قطع غیرطبیعی را به‌وسیله صدور اعلان اخطار خطای فرآیند نشان می‌دهد.

نتیجه اجرا و اعلانات اخطار خطای پردازش به‌وسیله نخ صادر شده و به مقصد(های) اعلان مناسب ارسال می‌شوند. در مورد یک لت راه‌اندازی غیرهمزمان، این اعلانات به مقصدهای اعلان خارجی فرستاده می‌شوند، در حالی که در مورد لت راه‌اندازی همزمان، این اعلانات برای لت راه‌اندازی منتشر می‌شوند.

1- System Management Scripting Language

مدیر می‌تواند به‌طور داوطلبانه تمام فرآیندهای راه‌اندازی را با استفاده از عملکرد حذف بر روی لت راه‌اندازی مربوط، متوقف کند. به‌محض دریافت عملکرد حذف، اگر انقیاد نام لت راه‌اندازی-نخ شامل تعریف "DELETE DELETES_CONTAINED-OBJECTS" باشد، همه نخ‌های آن لت راه‌اندازی که باعث اجرای اسکریپت شده‌اند، متوقف و حذف می‌شوند. سپس لت راه‌اندازی حذف می‌شود.

مدیر می‌تواند به‌طور داوطلبانه تمام اجراهای مربوط به یک نخ را به‌وسیله عملکرد حذف به نخ مربوط متوقف کند. به‌محض دریافت عملکرد حذف، اگر انقیاد نام نخ-نخ حاوی تعریف "DELETE DELETES-CONTAINED-OBJECTS" باشد، همه زیرنخ‌های مربوط به اجرای اسکریپت متوقف و حذف می‌شوند. سپس نخ حذف می‌شود.

برای خاتمه اجرای همه اسکریپت‌هایی که در حال حاضر به‌وسیله لت راه‌اندازی همزمان یا غیرهمزمان در حال اجرا هستند، می‌توان یک اقدام خاتمه به لت راه‌اندازی ارسال کرد. هنگامی که یک اقدام خاتمه به‌وسیله لت راه‌اندازی دریافت می‌شود، تمام نخ‌های مربوط به اجرای اسکریپت‌ها خاتمه یافته و حذف می‌شوند.

راه‌اندازی تمام نخ‌هایی که در حال حاضر به‌وسیله لت راه‌اندازی همزمان یا غیرهمزمان اجرا می‌شوند، ممکن است با ارسال یک اقدام معلق به لت راه‌اندازی ، معلق شود و سپس به‌وسیله یک اقدام از سرگیری، از سر گرفته شود.

اجرای اسکریپت‌ها به‌وسیله نخ‌های تعلیق‌پذیری که توسط یک لت راه‌اندازی غیرهمزمان ایجاد شده‌اند، ممکن است با ارسال یک اقدام تعلیق به نخ، تعلیق شده و در ادامه به‌وسیله یک اقدام ازسرگیری، از سر گرفته شوند. شناسه نخ برگردانده‌شده در اعلان نتیجه ماشه‌چکانی، باید به‌عنوان یک پارامتر به اقدامات تعلیق و ازسرگیری ارائه شود.

لت راه‌اندازی دارای صفاتی به‌منظور نظارت بر صفتی خاص از یک نمونه شیء خاص است. اگر مقدار صفت نظارت‌شده تغییرکند، یک ماشه‌چکانی برای اجرای فهرست اسکریپت مشخص‌شده، تولید می‌شود.

اگر صفت نظارت‌شده، یک فرق‌شمار رویداد (EDC)[1] مطابق آن‌چه که در ضمیمه پ تعریف شده باشد، اعلان‌های فیلترشده به‌وسیله EDC، راه‌اندازی اسکریپت‌ها را به‌وسیله لت راه‌اندازی ماشه‌چکانی می‌کنند.

۷-۳ مدیریت ترتیب‌سنج فرمان

مقادیر صفات لت راه‌اندازی، نخ، اسکریپت راه‌اندازی و نمونه‌های شیء مدیریت‌شده اسکریپت‌نویسی مخصوص، به ترتیب به‌وسیله عملیات Get و Set بازیابی شده و تغییر می‌کنند.

جداول ۱ تا ۵، صفات وضعیت ترتیب‌سنج فرمان، لت راه‌اندازی، نخ و اشیاء مدیریت‌شده اسکریپت را به وضعیت‌های تعریف‌شده در توصیه نامه CCITT Rec. X.731 | ISO/IEC 10164-2 نگاشت می‌کنند.

یادآوری- «-» به معنای هر مقدار دلخواه است.

1 -Event Discrimination Counter

جدول۱- جدول وضعیت ترتیب‌سنج فرمان

وضعیت ترتیب‌سنج فرمان	وضعیت مدیریتی	وضعیت عملیاتی
CS عملیاتی نیست.	-	غیرفعال شده
CS عملیاتی است.	قفل نشده	فعال شده
CS قفل شده است.	قفل شده	فعال شده
CS خاموش شده است.	خاموش شده	فعال شده

هنگامی‌که ترتیب‌سنج فرمان دارای وضعیت عملیاتی غیرفعال است، در وضعیت غیرقابل اجرای کامل است و لایه‌های راه‌اندازی آن، اسکریپت‌های در حال اجرا نیستند. اگر ترتیب‌سنج فرمان در حالت فعال باشد، یک رویداد متشکل از عملکردی که در محدوده شیء مدیریت‌شده انجام گرفته ، ممکن است منجر به یک انتقال از وضعیت مدیریتی قفل‌شده به وضعیت قفل‌نشده یا بالعکس شود. هنگامی‌که ترتیب‌سنج فرمان به یک وضعیت قفل‌شده وارد می‌شود لایه‌های راه‌اندازی آن، اجرای اسکریپت‌های راه‌اندازی را به حالت تعلیق در می‌آورند. از طرف دیگر، هنگامی‌که به یک وضعیت مدیریتی قفل‌نشده وارد می‌شود، لایه‌های راه‌اندازی برای شروع یا ازسرگیری اجرای اسکریپت‌های راه‌اندازی آماده هستند.

جدول۲- جدول وضعیت لت راه‌اندازی

وضعیت لت راه‌اندازی	وضعیت مدیریتی	وضعیت عملیاتی	وضعیت کنترل	وضعیت استفاده	وضعیت دسترسی
LP عملیاتی نیست.	-	غیرفعال شده	-	-	
LP عملیاتی است.	قفل نشده	فعال شده	-	مشغول	
LP عملیاتی است.	قفل نشده	فعال شده	-	بی‌کار	-
LP قفل شده است.	قفل شده	فعال شده	-	بی‌کار	-
LP در حال خدمت است.	-	-	-	-	خارج از خدمت نیست.
LP خارج از خدمت است.	-	-	-	-	خارج از خدمت
LP معلق شده است.	-	-	معلق	-	-
LP از سر گرفته شده است.	-	-	تهی	-	-

هنگامی که یک لت راه‌اندازی دارای وضعیت عملیاتی غیرفعال است، در وضعیت غیر قابل اجرای کامل قرار دارد و نمی‌تواند اسکریپت‌ها را اجرا کند. اگر در وضعیت فعال است، یک رویداد متشکل از عملکردی که در محدوده شیء مدیریت‌شده انجام شده ، ممکن است منجر به انتقال آن از وضعیت مدیریتی قفل‌شده به وضعیت قفل‌نشده یا بالعکس شود. هنگامی که لت راه‌اندازی به وضعیت قفل‌شده می‌رود، اجرای اسکریپت‌های راه‌اندازی را به حالت تعلیق در می‌آورد. از طرف دیگر، هنگامی که به وضعیت مدیریتی قفل‌نشده می‌رود، لایه‌های راه‌اندازی برای شروع و ازسرگیری اجرای اسکریپت‌های راه‌اندازی آماده هستند. لت راه‌اندازی مطابق با یک زمان‌بندی از پیش تعیین‌شده، به‌وسیله فرآیند کنترل داخلی غیرفعال می‌شود و مقدار وضعیت دسترس‌پذیری آن به خارج از خدمت تغییر می‌یابد.

یک اقدام تعلیق موجب می‌شود که وضعیت کنترل به تعلیق تغییر کند و یک اقدام ازسرگیری، آن را به مقدار پیش فرض، یعنی تهی باز می‌گرداند.

جدول ۳- جدول وضعیت نخ

وضعیت نخ	وضعیت عملیاتی
نخ عملیاتی نیست.	غیرفعال‌شده
نخ عملیاتی است.	فعال‌شده

هنگامی که نخ در حال انجام یک اجرای اسکریپت است، در وضعیت فعال‌شده قرار دارد، در غیر این‌صورت در وضعیت غیرفعال است.

جدول۴- جدول وضعیت نخ قابل تعلیق

وضعیت نخ تعلیق پذیر	وضعیت عملیاتی	وضعیت کنترل
نخ تعلیق‌پذیر عملیاتی نیست.	غیرفعال‌شده	-
نخ تعلیق‌پذیر عملیاتی است.	فعال‌شده	-
نخ تعلیق‌پذیر معلق شده است.	-	معلق
نخ تعلیق‌پذیر از سرگرفته شده است.	-	تهی

یک اقدام تعلیق منجر به تغییر وضعیت کنترل نخ تعلیق‌پذیر به تعلیق‌شده می‌شود و یک اقدام ازسرگیری آن را به مقدار پیش فرض، یعنی تهی باز می‌گرداند.

جدول ۵- جدول وضعیت اسکریپت راه‌اندازی

وضعیت اسکریپت راه‌اندازی	وضعیت مدیریتی
اجرای LS مجاز است	قفل نشده
اجرای LS مجاز نیست	قفل شده

یک رویداد که متشکل از عملکردی است که در محدوده شیء مدیریت‌شده انجام می‌شود، ممکن است منجر به انتقال اسکریپت از وضعیت مدیریتی قفل‌شده به وضعیت قفل‌نشده یا بالعکس شود. هنگامی که اسکریپت راه‌اندازی به وضعیت قفل‌شده می‌رود، نمی‌تواند به‌وسیله لت راه‌اندازی به جز لایه‌هایی که هم اکنون در حال اجرای آن هستند، اجرا شود. از طرف دیگر، هنگامی که به وضعیت مدیریتی قفل‌نشده می‌رود، اسکریپت راه‌اندازی برای اجرا شدن به‌وسیله سایر لایه‌های راه‌اندازی آماده است.

۷-۴ زمان‌بندی[1] ترتیب‌سنج فرمان

یک بسته زمان‌بندی زمان‌بند خارجی، قابلیت زمان‌بندی فعال‌سازی لایه‌های راه‌اندازی متعلق به ترتیب‌سنج فرمان را با استفاده از ماشه‌چکانی‌ها فراهم می‌کند. صفت وضعیت دسترس‌پذیری لت راه‌اندازی، مطابق با مشخصات زمان‌بندی تعیین‌شده به‌وسیله شیء مدیریت‌شده زمان‌بند خارجی به «خارج از خدمت» یا «در حال خدمت» تغییر می‌کند. مفاهیم بسته زمان‌بندی زمان‌بند خارجی در توصیه‌نامه‌های CCITT Rec. X.734 | ISO/IEC 10164-5 و CCITT Rec. X.735 | ISO/IEC 10164-6 شرح داده شده‌اند.

۷-۵ کنترل دسترسی

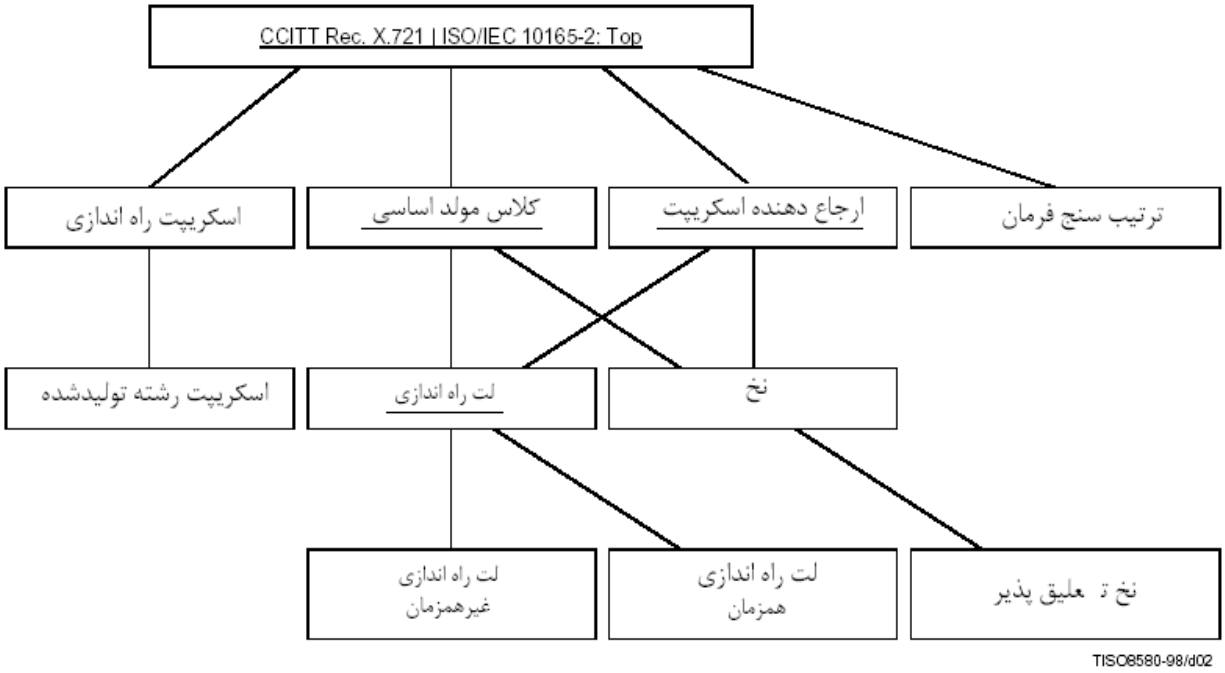
ترتیب‌سنج فرمان باید بتواند به کلیه نمونه‌های شیء مدیریت‌شده‌ای که به‌وسیله نخ‌های خود بر روی آن‌ها عمل می‌کند، دسترسی داشته باشد. هنگامی که یک عملکرد بر روی نمونه رد می‌شود، باید رفتار نخ در اسکریپت تعریف شده باشد. به‌عنوان مثال اگر یک عملکرد رد شود، ممکن است یک خطای تلاش برای دسترسی غیرمجاز به‌وسیله اعلان اخطار خطای پردازشی باز گردانده شود.

۸ تعاریف عمومی

۸-۱ اشیاء مدیریت‌شده[2]

این مشخصه، مجموعه‌ای از کلاس‌های شیء مدیریت‌شده را تعریف می‌کند. ساختار وراثتی این کلاس‌های شیء مدیریت‌شده در شکل ۲ نشان داده شده است.

1- Scheduling
2 - Managed objects



یادآوری – کلاس‌های شیء غیر قابل نمونه سازی[1] به‌صورت زیرخط دار نشان داده شده‌اند.

شکل ۲- ساختار وراثتی منابع ترتیب‌سنج فرمان

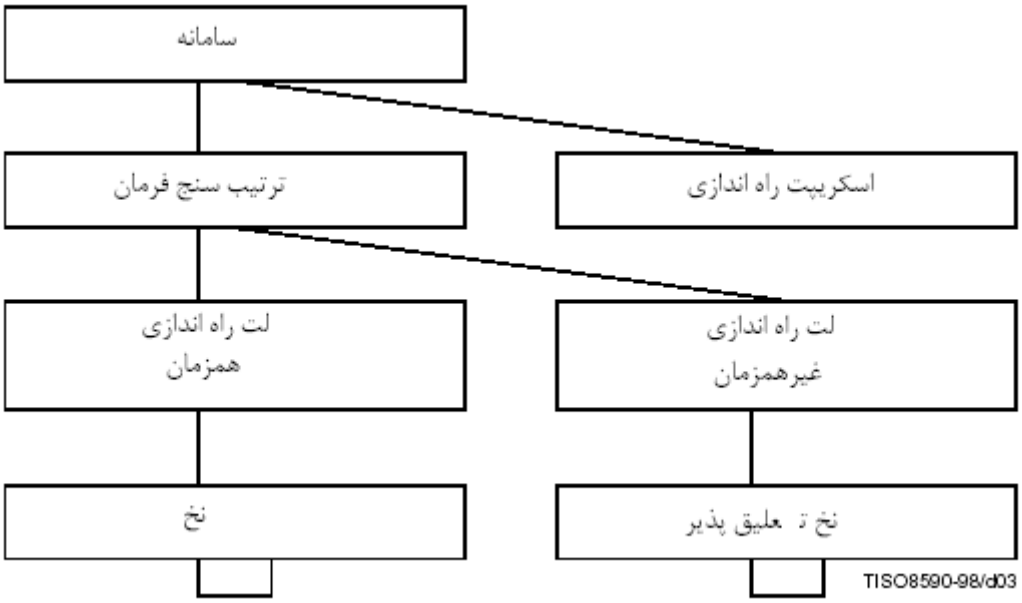
ساختار محتوای[2] این کلاس‌های شیء مدیریت‌شده در شکل ۳ نشان داده شده است.

۸-۱-۱ ترتیب‌سنج فرمان

۸-۱-۱-۱ مرور کلی

- یک شیء پشتیبانی مدیریت نشان‌دهنده منبعی که در نقش مدیر به‌عنوان فراخواننده عملیات تعیین‌شده به‌وسیله اسکریپت‌های راه‌اندازی آن و همچنین به‌عنوان یک مقصد اعلان، عمل می‌کند.
- به‌عنوان یک ارائه‌کننده خدمت برای یک لت راه‌اندازی عمل می‌کند.
- از یک یا چند لت راه‌اندازی تشکیل می‌شود.

1- Uninstantiable
2 -Containment



شکل۳- ساختار محتوای منابع ترتیب‌سنج فرمان

۸-۱-۱-۲ بسته‌های شیء پشتیبانی مدیریت ترتیب‌سنج فرمان

شیء پشتیبانی مدیریت ترتیب‌سنج فرمان دارای بسته اجباری زیر است:

- بسته ترتیب‌سنج فرمان.

۸-۱-۱-۳ مشخصات ترتیب‌سنج فرمان

کلاس شیء پشتیبانی مدیریت ترتیب‌سنج فرمان دارای صفات زیر است.

۸-۱-۱-۳-۱ شناسه ترتیب‌سنج فرمان

مقدار این صفت یک نمونه از کلاس شیء پشتیبانی مدیریت ترتیب‌سنج فرمان را تعریف می‌کند.

۸-۱-۱-۳-۲ حالت اداری[1]

این صفت نشان‌دهنده‌ی قابلیت اداری ترتیب‌سنج فرمان برای انجام کارکردش است. حالات اداری زیر تعریف شده‌اند:

الف) قفل‌نشده- لایه‌های راه‌اندازی ترتیب‌سنج فرمان مجاز به شروع و ازسرگیری اجرای اسکریپت‌های راه‌اندازی هستند.

ب) قفل‌شده- لایه‌های راه‌اندازی ترتیب‌سنج فرمان مجاز به شروع اجرای اسکریپت‌های راه‌اندازی نیستند. اگر اجراهایی در حال پیشرفت هستند، به حالت تعلیق در می‌آیند.

پ) خاموش شدن[1] - ترتیب‌سنج فرمان در حال خاموش شدن است و لایه‌های راه‌اندازی آن مجاز به انجام هیچ اجرای اسکریپتی نخواهند بود.

---

1 -administrative state

۸-۱-۱-۳-۳ وضعیت عملیاتی[۲]

این صفت نشان‌دهنده توانایی عملیاتی ترتیب‌سنج فرمان برای انجام عملکردش است.

وضعیت‌های عملیاتی زیر تعریف شده‌اند:

الف) فعال‌شده- ترتیب‌سنج فرمان عملیاتی و آماده استفاده است.

ب) غیرفعال‌شده- ترتیب‌سنج فرمان آماده استفاده نیست.

۸-۱-۱-۴اعلانات ترتیب‌سنج فرمان

کلاس شیء مدیریت پشتیبانی ترتیب‌سنج فرمان دارای اعلانات زیر است:

- ایجاد شیء، همان‌طورکه در توصیه‌نامه CCITT Rec. X.730 | ISO/IEC 10164-1 تعریف شده است.
- حذف شیء، همان‌طورکه در توصیه‌نامه CCITT Rec. X.730 | ISO/IEC 10164-1 تعریف شده است.
- تغییر حالت، همان‌طورکه در توصیه‌نامه CCITT Rec. X.730 | ISO/IEC 10164-1 تعریف شده است.

۸-۱-۲ نخ

۸-۱-۲-۱ مرور کلی

- یک شیء مدیریت که اجرای دستور را مدل‌سازی کرده و دستورات را بر طبق یک اسکریپت انجام می‌دهد.
- زیرکلاسی از مولد[۳] پایه و کلاس‌های شیء مدیریت‌شده‌ی ارجاع‌دهنده اسکریپت.
- هر نخ به یک لت راه‌اندازی همزمان اختصاص داده می‌شود.
- این کلاس شیء، یک اعلان دارد که نتیجه اجرای اسکریپت راه‌اندازی را اعلان می‌کند.

۸-۱-۲-۲ بسته‌های کلاس شیء مدیریت‌شده نخ

کلاس شیء مدیریت‌شده نخ دارای بسته‌های اجباری زیر است:

- بسته نخ؛
- بسته نتیجه اجرا.

۸-۱-۲-۳ مشخصات کلاس شیء مدیریت‌شده نخ

کلاس نخ دارای صفات زیر است.

۸-۱-۲-۳-۱ شناسه نخ

مقدار این صفت، یک نمونه از کلاس شیء مدیریت‌شده نخ را شناسایی می‌کند.

---

1 -Shutting down
2 -Operational state
3- Spawner

۸-۱-۲-۳-۲ شناسه اسکریپت

مقدار این صفت، یک نمونه از کلاس شیء مدیریت‌شده اسکریپت راه‌اندازی را که در حال اجرا است، شناسایی می‌کند.

۸-۱-۲-۳-۳ پارامترهای اجرایی[1]

مؤلفه پارامترهای اسکریپت این صفت، فهرستی از مقادیر پارامتر است که به‌وسیله یک لت راه‌اندازی که برای اجرای اسکریپت به آنها نیاز دارد، تهیه شده است. مؤلفه شناسه اسکریپت این صفت مشخص‌کننده اسکریپتی (اسکریپت‌هایی) است که باید با پارامترهای مربوط اجرا شوند.

۸-۱-۲-۳-۴ وضعیت عملیاتی[2]

این صفت نشان‌دهنده قابلیت عملیاتی نخ برای انجام کارکردش است. وضعیت‌های عملیاتی زیر تعریف شده‌اند:

الف) فعال‌شده- نخ عملیاتی است و در حال انجام یک اجرای اسکریپت است.

ب) غیرفعال‌شده- نخ عملیاتی نیست.

۸-۱-۲-۴ اعلانات کلاس شیء مدیریت‌شده نخ

کلاس شیء مدیریت‌شده نخ دارای اعلانات زیر است که آن‌ها را به کلاس شیء مدیریت‌شده لت راه‌اندازی ارسال می‌کند:

- نتیجه اجرا همان‌طورکه در زیربند ۸-۲-۱ تعریف شده است.
- اخطار خطای پردازش همان‌طورکه در CCITT X.733 | ISO/IEC 10164-4 تعریف شده است.

۸-۱-۳ نخ تعلیق‌پذیر[3]

۸-۱-۳-۱ مرور کلی

- زیرکلاسی از کلاس شیء مدیریت‌شده نخ.
- تولیدشده به‌وسیله یک لت راه‌اندازی غیرهمزمان.
- می‌تواند به‌وسیله اقدامات تعلیق و ازسرگیری، معلق شده یا از سرگرفته شود.

۸-۱-۳-۲ بسته‌های نخ تعلیق‌پذیر

کلاس شیء مدیریت‌شده نخ تعلیق‌پذیر دارای بسته اجباری زیر است:

- پذیرنده ازسرگیری تعلیق.

۸-۱-۳-۳ مشخصات نخ تعلیق‌پذیر

نخ تعلیق‌پذیر دارای صفات زیر است.

---

1 - Executing parameters
2 -Operational state
3 -Suspendable thread

۸-۱-۳-۳-۱ وضعیت کنترل

صفت کنترل وضعیت در توصیه‌نامه CCITT Rec. X.731 | ISO/IEC 10164-2 تعریف شده است. مقدار پیش‌فرض آن تهی است. اگر اجرا معلق شود، وضعیت کنترل به «معلق» تغییر می‌کند و اگر اجرا ازسر گرفته شود، مقدار آن به تهی تغییر می‌کند.

۸-۱-۳-۴ اعمال کلاس شیء مدیریت‌شده نخ تعلیق‌پذیر

اقدامات زیر می‌توانند به کلاس شیء مدیریت‌شده نخ تعلیق‌پذیر هدایت شوند:

- تعلیق؛
- از سرگیری.

۸-۱-۴ اسکریپت راه‌اندازی

۸-۱-۴-۱ مرور کلی

- اطلاعات مدیریتی که نشان‌دهنده دنباله‌ای از دستورالعمل‌ها در زبان‌های اسکریپت‌نویسی مخصوص هستند.
- هر اسکریپت باید مستقل باشد.

۸-۱-۴-۲ مشخصات اسکریپت راه‌اندازی

کلاس اسکریپت راه‌اندازی دارای صفات زیر را است.

۸-۱-۴-۲-۱ شناسه اسکریپت

مقدار این صفت، یک نمونه از کلاس شیء مدیریت‌شده اسکریپت راه‌اندازی را شناسایی می‌کند.

۸-۱-۴-۲-۲ نوع نتیجه اجرا

مقدار این صفت، نوع مورد انتظار نتیجه اجرا را شناسایی می‌کند.

۸-۱-۴-۲-۳ وضعیت مدیریتی

این مورد در توصیه نامه CCITT Rec. X.731 | ISO/IEC 10164-2 تعریف شده است.

۸-۱-۴-۳ بسته‌های کلاس شیء اسکریپت راه‌اندازی

کلاس شیء مدیریت‌شده اسکریپت راه‌اندازی دارای بسته اجباری زیر است:

- بسته اسکریپت راه‌اندازی.

۸-۱-۵ کلاس مولد پایه

۸-۱-۵-۱ مرور کلی

- بر قابلیت ایجاد نمونه‌های شیء مشمول جدید به همراه تولید نام خودکار برای آن‌ها، دلالت دارد.
- فوق‌کلاس[1] نخ ترتیب‌سنج فرمان و کلاس‌های شیء لت راه‌اندازی.

---

1 -Superclass

۸-۱-۵-۲ بسته‌های کلاس مولد پایه

کلاس مولد پایه دارای بسته اجباری زیر است:

- بسته مولد پایه.

۸-۱-۶ لت راه‌اندازی

۸-۱-۶-۱ مرور کلی

- زیرکلاسی از مولد پایه و کلاس‌های شیء مدیریت‌شده‌ی ارجاع‌دهنده‌ی اسکریپت.
- آغازگر اجرای اسکریپت راه‌اندازی هنگام دریافت یک ماشه‌چکانی.
- به‌عنوان یک IVMO برای یک نخ عمل می‌کند.
- یک اسکریپت با استفاده از یک یا چند نخ، به‌وسیله لت راه‌اندازی اجرا می‌شود.

۸-۱-۶-۲ بسته‌های کلاس شیء مدیریت‌شده لت راه‌اندازی

بسته‌های اجباری‌کلاس شیء مدیریت‌شده لت راه‌اندازی عبارتند از:

- بسته لت راه‌اندازی؛
- پذیرنده اقدام ماشه‌چکانی ؛
- ارسال‌کننده پارامتر؛
- بسته نتیجه ماشه‌چکانی؛
- پذیرنده رویداد ماشه‌چکانی؛
- پذیرنده خاتمه؛
- زمان‌بند خارجی؛
- پذیرنده ازسرگیری تعلیق.

۸-۱-۶-۳ مشخصات لت راه‌اندازی

کلاس شیء مدیریت‌شده لت راه‌اندازی دارای صفات زیر است.

۸-۱-۶-۳-۱ فهرست اسکریپت در دسترس

این فهرست، فهرستی از تمام اسکریپت‌هایی است که لت راه‌اندازی قادر به اجرای آنها است. لت راه‌اندازی فقط اسکریپت‌هایی را اجرا می‌کند که هم در مؤلفه فهرست پارامتر اجرایی از پارامترهای ماشه‌چکانی و هم در صفت فهرست اسکریپت قابل دسترس موجود باشند.

۸-۱-۶-۳-۲ فهرست پارامتر اجرایی پیش فرض

این فهرست، فهرستی از شناسه‌های اسکریپت و پارامترهای اسکریپت است که برای اجرای پیش فرض هنگامی‌که یک لت راه‌اندازی بدون پارامتر ماشه‌چکانی شود، استفاده می‌شود.

۸-۱-۶-۳-۳ وضعیت مدیریتی

این صفت، قابلیت مدیریتی لت راه‌اندازی را برای اجرای کارکردش نشان می‌دهد. وضعیت‌های مدیریتی زیر تعریف شده‌اند:

الف) قفل‌نشده - لت راه‌اندازی مجاز به شروع یا از سرگیری اجرای اسکریپت‌های راه‌اندازی است.

ب) قفل‌شده - لت راه‌اندازی مجاز به شروع اجرای اسکریپت‌های راه‌اندازی نیست. اگر اجراهایی در حال پیشرفت باشند، به حالت تعلیق درمی‌آیند.

۸-۱-۶-۳-۴ وضعیت عملیاتی

این صفت نشان‌دهنده قابلیت عملیاتی لت راه‌اندازی برای اجرای کارکرد آن است.

وضعیت‌های عملیاتی زیر تعریف شده‌اند:

الف) فعال - لت راه‌اندازی عملیاتی است و برای اجرای یک اسکریپت آماده است.

ب) غیرفعال - لت راه‌اندازی عملیاتی نیست و برای اجرای اسکریپت‌ها آماده نیست.

۸-۱-۶-۳-۵ وضعیت کاربری

این صفت نشان‌دهنده وضعیت کاربری لت راه‌اندازی است. وضعیت‌های کاربری زیر تعریف شده‌اند:

الف) مشغول[1] - لت راه‌اندازی برای اجرای یک اسکریپت راه‌اندازی استفاده می‌شود.

ب) بی‌کار[2] - لت راه‌اندازی برای اجرای یک اسکریپت استفاده نمی‌شود.

۸-۱-۶-۳-۶ وضعیت دسترس پذیری

شرایط این وضعیت نشان می‌دهد که آیا لت راه‌اندازی برای اجرای کارکرد خود در دسترس است. وضعیت‌های زیر تعریف شده‌اند:

الف) خارج از خدمت: لت راه‌اندازی به‌وسیله فرآیند کنترل داخلی، مطابق با یک برنامه زمانی از پیش تعین‌شده غیر فعال شده است.

ب) غیر خارج از خدمت: صفت وضعیت دسترس‌پذیری دارای مقدار خارج از خدمت نیست و در نتیجه فعال شده است.

۸-۱-۶-۳-۷ نمونه شیء مشاهده شده

این مورد در ITU-T Rec. X.739 | ISO/IEC 10164-11 تعریف شده است.

۸-۱-۶-۳-۸ شناسه صفت مشاهده شده

این مورد در ITU-T Rec. X.739 | ISO/IEC 10164-11 تعریف شده است.

۸-۱-۶-۳-۹ وضعیت کنترل

این صفت در CCITT Rec. X.731 | ISO/IEC 10164-2 تعریف شده است. مقدار پیش فرض تهی است. اگر اجرا به حالت تعلیق درآید، وضعیت کنترل به «معلق» تغییر می‌یابد و اگر اجرا از سر گرفته شود، مقدار به تهی تغییر می‌یابد.

---

1 -Busy
2- Idle

۸-۱-۶-۳-۱۰ شناسه لت راه‌اندازی

این صفت یک نمونه از کلاس شیء مدیریت‌شده لت راه‌اندازی را نام‌گذاری می‌کند.

۸-۱-۶-۴ اعلان‌های کلاس شیء مدیریت‌شده لت راه‌اندازی

کلاس شیء مدیریت‌شده لت راه‌اندازی دارای اعلان‌های زیر است که می‌توانند به مقصد(های) اعلان مناسب فرستاده شوند:

- نتیجه ماشه‌چکانی همان‌گونه که در زیربند ۸-۲-۱ تعریف شده است.
- هشدار خطای فرآیند همان‌گونه که در CCITT X.733 | ISO/IEC 10164-4 تعریف شده است.

۸-۱-۷ لت راه‌اندازی غیرهمزمان

۸-۱-۷-۱ مرور کلی

- زیر کلاسی از کلاس شیء مدیریت‌شده لت راه‌اندازی.
- نخ‌های قابل تعلیق به‌وسیله لت راه‌اندازی غیرهمزمان تولید می‌شوند.
- به‌محض اینکه تمام نخ‌های قابل تعلیق تولید شوند، یک اعلان نتیجه ماشه‌چکانی ارسال می‌کند.

۸-۱-۷-۲ بسته‌های لت راه‌اندازی غیرهمزمان

۸-۱-۷-۳ کلاس شیء مدیریت‌شده

کلاس شیء مدیریت‌شده لت راه‌اندازی غیرهمزمان دارای بسته زیر است:

- بسته نتیجه غیرهمزمان ماشه‌چکانی.

۸-۱-۸ لت راه‌اندازی همزمان

۸-۱-۸-۱ مرور کلی

- زیر کلاسی از کلاس شیء مدیریت‌شده لت راه‌اندازی.
- نخ‌ها به‌وسیله لت راه‌اندازی همزمان تولید می‌شوند.
- به‌محض اینکه تمام نخ‌ها تکمیل شوند، یک اعلان نتیجه ماشه‌چکانی پس از همگام‌سازی نتایج ارسال می‌کند.

۸-۱-۸-۲ بسته‌های کلاس شیء مدیریت‌شده لت راه‌اندازی همزمان

کلاس شیء مدیریت‌شده لت راه‌اندازی غیرهمزمان دارای بسته زیر است:

- بسته نتیجه همزمان ماشه‌چکانی.

۸-۱-۹ اسکریپت رشته عمومی

۸-۱-۹-۱ مرور کلی

- زیر کلاسی از کلاس شیء لایه راه‌اندازی.
- اسکریپت‌هایی که می‌توانند به شکل یک رشته عمومی نشان داده شوند.
- ممکن است اسکریپت‌های مخصوص دیگری به‌عنوان زیر کلاس اضافه شوند.

۸-۱-۹-۲ مشخصات زبان اسکریپت‌نویسی رشته عمومی

کلاس شیء مدیریت‌شده به‌وسیله زبان اسکریپت‌نویسی رشته عمومی دارای صفات زیر است.

۸-۱-۹-۲-۱ نام زبان اسکریپت‌نویسی

نام زبانی است که ویژگی‌های نحوی و معنایی یک اسکریپت را که به‌عنوان رشته عمومی نشان داده می‌شود، تعریف می‌کند.

۸-۱-۹-۲-۲ محتوای اسکریپت

به اسکریپتی که به‌عنوان یک رشته عمومی بیان شده است، اشاره می کند.

۸-۱-۹-۳ بسته‌های زبان اسکریپت‌نویسی رشته عمومی

کلاس شیء مدیریت‌شده زبان اسکریپت‌نویسی رشته عمومی‌دارای بسته اجباری زیر است:

- بسته اسکریپت رشته عمومی.

۸-۱-۱۰ ارجاع‌دهنده اسکریپت

۸-۱-۱۰-۱ مرور کلی

- فوق‌کلاس برای کلاس‌های شیء مدیریت‌شده لت راه‌اندازی و نخ.
- یک نگاشت رابطه ارجاع بین لت راه‌اندازی و اسکریپت راه‌اندازی و نخ و لت راه‌اندازی تعریف می‌کند.

۸-۱-۱۰-۲ بسته‌های ارجاع‌دهنده اسکریپت

ارجاع‌دهنده‌ی اسکریپت دارای بسته اجباری زیر است:

- بسته ارجاع دهنده اسکریپت.

۸-۲ اعلان‌های عمومی

اعلان‌های زیر در این استاندارد تعریف شده‌اند.

۸-۲-۱ نتیجه ماشه‌چکانی

این اعلان نتیجه اجرای یک اسکریپت را باز می‌گرداند، که در آن مقدار errorCode در صورت موفقیت برابر noError و در غیر این‌صورت برابر یک کد خطای مناسب برای نشان دادن ماهیت شکست، قرار داده می‌شود. کد خطایی که می‌تواند در فیلد errorCode از اعلان executionResult برگردانده شود عبارت است از:

- بدون خطا، در صورتی که اجرا موفقیت‌آمیز باشد؛
- خطای عدم اسکریپت، اگر اجرا شکست خورده باشد، چرا که نام اسکریپت مشخص نشده است؛
- خطای رد اسکریپت، اگر اسکریپت در فهرست اسکریپت‌هایی که لت راه‌اندازی برای اجرای آن پیکربندی شده است، نباشد؛
- خطای نوع پارامتر نامعتبر، درصورت عدم تطابق نوع پارامتر مورد انتظار اسکریپت و آن‌چه به‌وسیله آن اسکریپت ارائه شده است؛
- خطای مقدار پارامتر نامعتبر، اگر مقدار عرضه‌شده در پارامتر نامعتبر باشد، به‌عنوان مثال خارج از محدوده؛

- خطای نحو اسکریپت، اگر اجرای اسکریپت به دلیل یک خطای نحو در اسکریپت با شکست مواجه شده باشد؛
- خطای شکست اجرای اسکریپت، اگر اجرای اسکریپت به دلایلی غیر از نحو نادرست با شکست مواجه شده باشد؛
- تعداد پارامتر نامعتبر، اگر تعداد پارامترهای ارائه‌شده با تعداد پارامترهای مورد انتظار اسکریپت، ناسازگار باشند؛
- خطای دسترسی غیرمجاز، درصورتی‌که دسترسی به یک یا چند نمونه شیء که به‌وسیله اسکریپت استفاده می‌شوند، رد شود.

۸-۲-۲ نتیجه اجرا

نتیجه اجرای یک اسکریپت را باز می‌گرداند و اگر اجرا موفقیت‌آمیز باشد، مقدار errorCode برابر noError و در غیر این‌صورت برابر یک کد خطای مناسب برای نشان دادن ماهیت شکست، قرار داده می‌شود. کد خطایی که می‌تواند در فیلد errorCode از اعلان executionResult بازگردانده شود عبارت است از:

- بدون خطا، در صورت اجرای موفقیت‌آمیز؛
- خطای عدم اسکریپت، اگر اجرا شکست خورده باشد، چرا که نام اسکریپت مشخص نشده است ؛
- خطای رد اسکریپت، اگر اسکریپت در فهرست اسکریپت‌هایی که لت راه‌اندازی برای اجرای آن پیکربندی شده است، نباشد؛
- خطای نوع پارامتر نامعتبر، درصورت عدم تطابق نوع پارامتر مورد انتظار اسکریپت و آن‌چه به‌وسیله آن اسکریپت ارائه شده است؛
- خطای مقدار پارامتر نامعتبر، اگر مقدار عرضه‌شده در پارامتر نامعتبر باشد، به‌عنوان مثال خارج از محدوده؛
- خطای نحو اسکریپت، اگر اجرای اسکریپت به دلیل یک خطای نحو در اسکریپت با شکست مواجه شده باشد؛
- خطای شکست اجرای اسکریپت، اگر اجرای اسکریپت به دلایلی غیر از نحو نادرست با شکست مواجه شده باشد؛
- تعداد پارامتر نامعتبر، اگر تعداد پارامترهای ارائه‌شده با تعداد پارامترهای مورد انتظار اسکریپت، ناسازگار باشند؛
- خطای دسترسی غیرمجاز، درصورتی‌که دسترسی به یک یا چند نمونه شیء که به‌وسیله اسکریپت استفاده می‌شوند، پذیرفته نشود.

۸-۳ اقدامات عمومی

انواع اقدامات زیر در این استاندارد تعریف شده‌اند. این اقدامات برای کلاس‌های شیء مدیریت‌شده لت راه‌اندازی و نخ قابل تعلیق در این استاندارد تعریف شده‌اند.

۸-۳-۱ اقدام تعلیق

اقدام تعلیق ارسال‌شده به یک شیء مدیریت‌شده باعث می‌شود که آن شیء اجرای اسکریپتی را که هم اکنون در حال اجرای آن است، به حالت تعلیق در آورد. پارامترهایی که به‌وسیله اقدام تعلیق حمل می‌شوند، عبارتند از شناسه ماشه‌چکانی برای شناسایی این اقدام، یا شناسه نخ، اگر اقدام به یک نخ ارسال شده باشد، یا شناسه لت راه‌اندازی، اگر اقدام به یک لت راه‌اندازی ارسال شده باشد.

۸-۳-۲ اقدام ازسرگیری

اقدام ازسرگیری که به یک شیء مدیریت‌شده ارسال شده است، باعث می‌شود که آن شیء اجرای اسکریپتی را که به‌وسیله یک اقدام تعلیق قبلی به حالت تعلیق در آمده است، از سرگیرد. پارامترهایی که به‌وسیله اقدام تعلیق حمل می‌شوند، عبارتند از شناسه ماشه‌چکانی برای شناسایی این اقدام، یا شناسه نخ، اگر اقدام به یک نخ ارسال شده باشد، یا شناسه لت راه‌اندازی، اگر اقدام به یک لت راه‌اندازی ارسال شده باشد.

۸-۳-۳ اقدام خاتمه

اقدام خاتمه که به یک شیء مدیریت‌شده ارسال می‌شود، باعث خاتمه بی قید و شرط اجرای هر اسکریپتی به‌وسیله آن شیء، و هر شیء تولیدشده به‌وسیله آن شیء می‌شود. پارامتر شناسه ماشه‌چکانی برای شناسایی این اقدام استفاده می‌شود.

۸-۳-۴ اقدام ماشه‌چکانی

یک اقدام ماشه‌چکانی باعث آغاز اجرای اسکریپت می‌شود. این اقدام، پارامتر شناسه ماشه‌چکانی را فراهم می‌کند که این اقدام و مؤلفه فهرست پارامتر اجرای آن را شناسایی می‌کند که از دنباله‌ای از شناسه‌های اسکریپت تشکیل شده است که اسکریپت‌هایی را که باید اجرا شوند، به‌همراه پارامترهای اسکریپتی که برای اجرای این اسکریپت‌ها مورد نیاز هستند، مشخص می‌کند.

۹ خدمات

۹-۱ مقدمه

ترتیب‌سنج فرمان خدماتی را برای تغییر اقدامات ترتیب‌سنج فرمان و لایه‌های راه‌اندازی ارائه می‌دهد. به‌طور خاص، اقداماتی که می‌توانند برای هر نمونه از ترتیب‌سنج فرمان و اسکریپت‌های راه‌اندازی به‌کار روند عبارتند از:

- ایجاد نمونه‌هایی از ترتیب‌سنج فرمان، لت راه‌اندازی و اسکریپت راه‌اندازی؛
- حذف نمونه‌های لت راه‌اندازی و اسکریپت راه‌اندازی ؛
- تغییر صفات ترتیب‌سنج فرمان، لت راه‌اندازی و اسکریپت راه‌اندازی؛
- بازیابی صفات ترتیب‌سنج فرمان، لت راه‌اندازی و اسکریپت راه‌اندازی.

این کارکرد، علاوه بر موارد فوق خدماتی را برای تغییر نمونه‌ها، خدمات اعلان و اقدامی برای ماشه‌چکانی اجرای فرمان، ارائه می‌کند.

۹-۲ خدمات آغاز، خاتمه، تغییر و بازیابی

خدمات PT-CREATE، PT-DELETE، PT-SET و PT-GET می‌توانند برای ایجاد، حذف، تغییر و بازیابی مقادیر صفات نمونه‌های شیء پشتیبانی مدیریت ترتیب‌سنج فرمان و شیء مدیریت‌شده اسکریپت راه‌اندازی و لت راه‌اندازی مورد استفاده قرار گیرند.

۹-۳ خدمات اعلان

۹-۳-۱ تعریف خدمت نتیجه اجرا

این زیربند، خدمت گزارش executionResultInfo را مشخص می‌کند که در این استاندارد ملی تعریف شده است و آن را به خدمت CMIS M-EVENT-REPORT نگاشت می‌کند.

جدول ۶ – پارامترهای گزارش نتیجه اجرا

نام پارامتر	Req/Ind	Rsp/Cnf
شناسه فراخوانی	P	P
حالت	P	-
کلاس شیء مدیریت شده	P	P
نمونه شیء مدیریت شده	P	P
نوع رویداد	M	C (=)
زمان رویداد	P	-
اطلاعات رویداد		
شناسه ماشه‌چکانی	M	-
شناسه اسکریپت	M	-
شناسه نخ	M	-
کد خطا	U	-
نوع نتیجه اجرا	U	-
نتیجه اجرا	U	-
زمان کنونی	-	P
پاسخ رویداد	-	P
خطاها	-	P

جدول ۷ – پارامترهای گزارش نتیجه ماشه‌چکانی

نام پارامتر	Req/Ind	Rsp/Cnf
شناسه فراخوانی	P	P
حالت	P	-
کلاس شیء مدیریت شده	P	P
نمونه شیء مدیریت شده	P	P
نوع رویداد	M	C(=)
زمان رویداد	P	-

ادامه جدول ۷

اطلاعات رویداد		
شناسه ماشه‌چکانی	M	-
شناسه اسکریپت	U	-
شناسه نخ	-	-
پارامترهای اسکریپت	U	-
نوع اجرا	U	-
کد خطا	U	-
نتیجه اجرا	U	-
زمان کنونی	-	P
پاسخ رویداد	-	P
خطاها	-	P

۹-۴ خدمات اقدام

این زیربند، خدمات اقدام ماشه‌چکانی و خاتمه را که در این استاندارد ملی تعریف شده‌اند، مشخص می‌کند و آنها را به خدمت CMIS M-EVENT-ACTION نگاشت می‌دهد.

جدول ۸ – پارامترهای خدمت اقدام ماشه‌چکانی

نام پارامتر	Req/Ind	Rsp/Conf
شناسه فراخوانی	P	P
شناسه متصل	-	P
حالت	P	-
کلاس شیء پایه	P	-
نمونه شیء پایه	P	-
دامنه کاربرد	P	-
فیلتر	P	-
کلاس شیء مدیریت شده	-	P
نمونه شیء مدیریت شده	-	P
کنترل دسترسی	P	-
همگام‌سازی	P	-
نوع اقدام	M	-
اطلاعات اقدام		
شناسه ماشه‌چکانی	M	-
فهرست ویژگی	U	-
خطاها	-	P

جدول ۹ – پارامترهای خدمت اقدام خاتمه

نام پارامتر	Req/Ind	Rsp/Conf
شناسه فراخوانی	P	P
شناسه متصل	-	P
حالت	P	-
کلاس شیء پایه	P	-
نمونه شیء پایه	P	-
دامنه کاربرد	P	-
فیلتر	P	-
کلاس شیء مدیریت شده	-	P
نمونه شیء مدیریت شده	-	P
کنترل دسترسی	P	-
همگام‌سازی	P	-
نوع اقدام	M	-
اطلاعات اقدام		
شناسه ماشه‌چکانی	M	-
خطاها	-	P

جدول ۱۰ – پارامترهای خدمت اقدام تعلیق

نام پارامتر	Req/Ind	Rsp/Conf
شناسه فراخوانی	P	P
شناسه متصل	-	P
حالت	P	-
کلاس شیء پایه	P	-
نمونه شیء پایه	P	-
دامنه کاربرد	P	-
فیلتر	P	-
کلاس شیء مدیریت شده	-	P
نمونه شیء مدیریت شده	-	P
کنترل دسترسی	P	-
همگام‌سازی	P	-
نوع اقدام	M	-
اطلاعات اقدام		
شناسه ماشه‌چکانی	M	-
شناسه نخ	U	-
شناسه لت راه‌اندازی	U	-
خطاها	-	P

جدول ۱۱ – پارامترهای خدمت اقدام ازسرگیری

نام پارامتر	Req/Ind	Rsp/Conf
شناسه فراخوانی	P	P
شناسه متصل	-	P
حالت	P	-
کلاس شیء پایه	P	-
نمونه شیء پایه	P	-
دامنه کاربرد	P	-
فیلتر	P	-
کلاس شیء مدیریت شده	-	P
نمونه شیء مدیریت شده	-	P
کنترل دسترسی	P	-
همگام‌سازی	P	-
نوع اقدام	M	-
اطلاعات اقدام		
شناسه ماشه‌چکانی	M	-
شناسه نخ	U	-
شناسه لت راه‌اندازی	U	-
خطاها	-	P

۱۰ واحدهای کارکردی

سه واحد کارکردی در این استاندارد ملی برای مدیریت ترتیب‌سنج‌های فرمان تعریف شده است:

الف) واحد کارکردی اجرایی : واحد کارکردی اجرایی نیاز به خدمات PT-CREATE، خدمت اعلان نتیجه اجرای اقدام ماشه‌چکانی و خدمت گزارش هشدار خطای پردازش دارد.

ب) واحد کارکردی نظارت : واحد کارکردی نظارت نیاز به خدمات PT-GET دارد.

پ) واحد کارکردی کنترل : واحد کارکردی کنترل نیاز به خدمات اقدام خاتمه، PT-DELETE دارد.

۱۱ پروتکل‌ها و نحو انتزاعی

۱۱-۱ نحو انتزاعی

۱۱-۱-۱ اشیاء مدیریت‌شده

۱۱-۱-۱-۱ اشیاء مدیریت‌شده تعریف‌شده

جدول ۱۲ رابطه بین اشیاء مدیریت‌شده تعریف‌شده در زیربند ۸-۱ و مشخصه کلاس شیء مدیریت‌شده در ضمیمه الف را شناسایی می‌کند.

جدول ۱۲ – اشیاء مدیریت شده و برچسب‌های ارجاع

نام شیء مدیریت شده	برچسب ارجاع
مولد پایه	basicSpawnerClass
نخ تعلیق‌پذیر	suspendableThread
نخ	thread
لت راه‌اندازی غیر همزمان	asynchronousLaunchPad
لت راه‌اندازی همزمان	synchronousLaunchPad
لت راه‌اندازی	launchPad
اسکریپت راه‌اندازی	launchScript
ترتیب‌سنج فرمان	commandSequencer
اسکریپت رشته عمومی	generalStringScript
ارجاع دهنده اسکریپت	scriptReferencer

۱۱-۲ صفات

۱۱-۲-۱ صفات واردشده از تعریف اطلاعات مدیریت

این استاندارد ، به صفات مدیریتی زیر ارجاع می‌کند، که نحو انتزاعی آن‌ها در CCITT Rec. X.721 | ISO / IEC 10165-2 مشخص شده است:

الف) حالت مدیریتی؛

ب) حالت کاربری؛

پ) حالت عملیاتی؛

ت) وضعیت کنترل؛

ث) وضعیت دسترس‌پذیری؛

این استاندارد همچنین به صفات مدیریتی زیر ارجاع می‌کند، که نحو انتزاعی آن در ITU-T Rec. X.739 | ISO/IEC 10164-11 مشخص شده است:

الف) نمونه شیء مشاهده‌شده؛

ب) شناسه صفت مشاهده‌شده.

۱۱-۲-۲ صفات تعریف‌شده در این استاندارد

این استاندارد ، صفات مدیریتی زیر را تعریف می‌کند، که نحو انتزاعی آن‌ها در ضمیمه الف مشخص شده است:

الف) شناسه لت راه‌اندازی ؛

ب) شناسه ترتیب‌سنج فرمان؛

ج) شناسه نخ؛

د) شناسه اسکریپت؛

ه) شناسه ماشه‌چکانی ؛

و) نوع نتیجه اجرا؛

ز) پارامترهای اجرا؛

ح) نام زبان اسکریپت؛

ط) محتوای اسکریپت؛

ی) فهرست پارامتر اجرایی پیش فرض؛

ک) فهرست اسکریپت قابل دسترس؛

۱۱-۲-۳ نگاشت پارامتر به صفت

جدول ۳ رابطه بین پارامترهای خدمت تعریف‌شده در زیربندهای ۸-۱ و ۸-۲ و مشخصات نوع صفت در پیوست الف را شناسایی می‌کند.

جدول ۱۳ – نام‌های پارامترها و صفات

نام پارامتر	نام صفت
شناسه لت راه‌اندازی	Launch pad id
شناسه ترتیب سنج فرمان	Command sequencer id
شناسه نخ	Thread id
شناسه اسکریپت	Script id
شناسه ماشه‌چکانی	Trigger id
نوع نتیجه اجرا	Execution result type
پارامترهای اجرا	Executing parameters
نام زبان اسکریپت	Script language name
محتوای اسکریپت	Script content
فهرست پارامتر اجرای پیش فرض	Default execution parameter list
فهرست اسکریپت قابل دسترس	Available script list

۱۱-۳ خالی

۱۱-۴ اعلان‌ها

۱۱-۴-۱ اعلان‌های ارجاع‌شده

این استاندارد ، به رویدادهای تعریف‌شده در CCITT Rec. X.730 | ISO/IEC 10164-1 ارجاع می‌کند:

الف) اعلان ایجاد شیء؛

ب) اعلان حذف شیء؛

ج) اعلان هشدار خطای پردازش.

این اســتانـدارد ، همچنین به رویداد زیر که در CCITT Rec. X.731 | ISO/IEC 10164-2 تعریف شده‌اند، ارجاع می‌کند؛

- اعلان تغییر حالت.

۱۱-۴-۲ اعلان‌های تعریف شده در این استاندارد

جدول ۱۴ رابطه بین اعلان‌های تعریف‌شده در زیربند ۹-۳ و مشخصات نوع اعلان در پیوست الف را شناسایی می‌کند.

جدول ۱۴ – اعلان‌ها

نوع رویداد	نوع اعلان
نتیجه ماشه‌چکانی	triggerResultInfo
نتیجه اجرا	executionResultInfo

۱۱-۵ اقدامات

۱۱-۵-۱ اقدامات تعریف‌شده در این استاندارد

جدول ۱۵ رابطه بین اقدامات تعریف‌شده در زیربند ۹-۴ و مشخصات نوع اعلان در **پیوست** الف را شناسایی می‌کند.

جدول ۱۵ – اقدامات

نام اقدام	برچسب مرجع
خاتمه	terminate
تعلیق	suspend
ازسرگیری	resume
ماشه‌چکانی	trigger

۱۱-۶ مذاکره واحدهای کارکردی

این استاندارد ملی، مقدار شناسه شیء زیر را به‌عنوان یک مقدار از نوع ASN.1 به نام FunctionalUnitPackageId که در CCITT Rec. X.701 | ISO / IEC 10040 تعریف شده است، اختصاص می‌دهد:

{joint-iso-itu-t ms(9) function(2) part21(21) functionalUnitPackage(1)}

این مقدار برای مذاکره واحدهای کارکردی زیر استفاده می‌شود:

0 واحد کارکردی اجرایی

M واحد کارکردی نظارتی

2 واحد کارکردی کنترلی

که شماره، موقعیت‌های بیتی را در رشته بیتی اختصاص‌یافته به واحدهای کارکردی شناسایی می‌کند، و نام‌های ارجاع‌کننده به واحدهای کارکردی در بند ۱۰ تعریف شده‌اند.

در محتوای برنامه کاربردی مدیریت سامانه‌ها، سازوکار مذاکره واحدهای کارکردی به‌وسیله CCITT Rec. X.701 | ISO/ IEC 10040 توصیف شده است.

یادآوری- نیاز به مذاکره واحدهای کارکردی به‌وسیله محتوای برنامه کاربردی مشخص می‌شود.

۱۲ رابطه با دیگر کارکردها

ترتیب‌سنج فرمان، خدمات تعریف‌شده در CCITT Rec. X.731 | ISO/ IEC 10164-2 را برای اعلان تغییرات حالت، خدمات تعریف‌شده در CCITT Rec. X.730 | ISO/ IEC 10164-1 را برای ایجاد و حذف اشیاء مدیریت‌شده، بازیابی صفات و اعلان تغییرات مقدار صفات مورد استفاده قرار می‌دهد.

ترتیب‌سنج فرمان از خدمات تعریف‌شده در ITU-T Rec. X.741 | ISO/IEC 10164-9 برای ارائه قابلیت‌های کنترل دسترسی به نمونه‌های شیء مدیریت‌شده‌ای که نخ‌ها می‌توانند بر روی آن‌ها عمل کنند، استفاده می‌کند.

۱۳ انطباق

دو کلاس انطباق وجود دارد: کلاس انطباق عمومی[1] و کلاس انطباق وابسته[2]. یک سامانه که ادعای پیاده‌سازی عناصر رویه‌ای را برای خدمات مدیریت سامانه‌ها که به‌وسیله این مشخصه ارجاع شده‌اند، دارد باید الزامات کلاس انطباق عمومی یا وابسته را، همان‌طورکه در زیربندهای زیر تعریف شده است، برآورده سازد.

تهیه‌کننده پیاده‌سازی باید کلاس انطباق مورد ادعا را بیان کند.

۱۳-۱ الزامات کلاس انطباق عمومی

یک سامانه که ادعای انطباق عمومی را دارد باید این کارکرد را برای تمام کلاس‌های شیء مدیریت‌شده‌ای که اطلاعات مدیریتی تعریف‌شده در این استاندارد را وارد می‌کنند، پشتیبانی کند.

یادآوری- این مورد برای همه زیرکلاس‌ها از کلاس‌های شیء پشتیبان مدیریت که در این استاندارد تعریف شده‌اند، کاربردپذیر است.

۱۳-۱-۱ انطباق ایستا

سامانه باید:

الف) از نقش مدیر یا عامل یا هر دو، با توجه به واحد کارکردی معیارهای کنترل[3] و واحد کارکردی معیارهای نظارت[4]، پشتیبانی کند؛

1- General conformance
2- Dependent conformance
3- Control metrics
4 -Monitor metrics

ب) از نحو انتقال مشتق‌شده از قواعد کدبندی مشخص‌شده در CCITT Rec. X. 209 | ISO/IEC 8825 به نام {joint-iso-itu-t asn1(1) basicEncoding(1)}، به‌منظور تولید و تفسیر MAPDUها که به‌وسیله نوع داده‌های انتزاعی ارجاع‌شده در زیربندهای ۱۱-۴ و ۱۱-۵ تعریف شده‌اند، پشتیبانی کند.

ج) هنگامی که در نقش عامل عمل می‌کند، حداقل از یک یا چند نمونه از ترتیب‌سنج فرمان، لت راه‌اندازی، اسکریپت راه‌اندازی، کلاس‌های شیء مدیریت‌شده نخ یا هر یک از زیرکلاس‌های آن‌ها پشتیبانی کند.

۱۳-۱-۲ انطباق پویا

سامانه باید در نقش(هایی) که برای آن‌ها ادعای انطباق کرده است:

الف) از عناصر رویه تعریف‌شده در موارد زیر پشتیبانی کند:

- CCITT Rec. X.730 | ISO/ IEC 10164-1 برای خدمات PT-GET ، PT-CREATE، PT-DELETE، PT-SET، گزارش ایجاد شیء، گزارش حذف شیء و گزارش تغییر صفت؛
- CCITT Rec. X.731 | ISO/ IEC 10164-2 برای خدمت گزارش تغییر حالت.
- CCITT Rec. X.733 | ISO/ IEC 10164-4 برای خدمت گزارش هشدار خطای پردازش.

ب) از عناصر رویه تعریف‌شده در این استاندارد برای خدمات گزارش و اقدام زیر پشتیبانی کند:

- اعلان نتیجه اجرا؛
- اقدام ماشه‌چکانی؛
- اقدام خاتمه؛
- اقدام تعلیق؛
- اقدام از سر گیری.

۱۳-۲ الزامات کلاس انطباق وابسته

۱۳-۲-۱ انطباق ایستا

سامانه باید:

الف) از نحو انتقال مشتق‌شده از قواعد کد بندی مشخص‌شده در CCITT Rec. X.209 | ISO/ IEC 8825 به نام {joint-iso-itu-t asn1(1) basicEncoding(1)}، به‌منظور تولید و تفسیر MAPDUها که به‌وسیله انواع داده انتزاعی ارجاع‌شده در زیربند ۱۱-۱ تعریف شده‌اند، همان‌گونه‌که مورد نیاز یک استاندارد مرجع است، پشتیبانی کند؛

ب) هنگامی که در نقش عامل عمل می‌کند، از یک یا چند نمونه از موارد ترتیب‌سنج فرمان، لت راه‌اندازی، اسکریپت راه‌اندازی، کلاس‌های شیء مدیریت‌شده نخ یا هر یک از زیرکلاس‌های آن‌ها، پشتیبانی کند.

۱۳-۲-۲ انطباق پویا

سامانه باید از عناصر رویه ارجاع‌شده به‌وسیله این استاندارد ، همان‌گونه‌که مورد نیاز یک استاندارد مرجع است، پشتیبانی کند.

۱۳-۳ انطباق برای پشتیبانی از تعاریف شیء مدیریت‌شده

اشیاء ترتیب‌سنج فرمان که به‌وسیله سامانه باز پشتیبانی می‌شوند، باید با رفتار مشخص‌شده در بند ۸ و نحو مشخص‌شده در پیوست الف، مطابقت داشته باشند.

پیوست الف

(الزامی)

تعریف اطلاعات مدیریتی

الف-۱ تعاریف کلاس شیء مدیریت شده

الف-۱-۱ اشیاء پایه

```
basicSpawnerClass MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721 | ISO/IEC 10165-2:1992":top;
CHARACTERIZED BY basicSpawnerPackage ;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx1(1)};

commandSequencer MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721 | ISO/IEC 10165-2:1992":top;
CHARACTERIZED BY
        commandSequencerPackage PACKAGE
        BEHAVIOUR commandSequencerBehaviour BEHAVIOUR
        DEFINED AS "An instance of this class represents a resource acting in a manager
        role as an invoker of operations determined by its launch scripts.";;
        ATTRIBUTES
                commandSequencerId GET,
        "CCITT Rec. X.731|ISO/IEC 10164-2:1992": administrativeState GET-REPLACE,
                "CCITT Rec. X.731|ISO/IEC 10164-2:1992":operationalState GET;
        NOTIFICATIONS
                "CCITT Rec. X.730 | ISO/IEC 10164-1": objectCreation,
                "CCITT Rec. X.730 | ISO/IEC 10164-1": objectDeletion,
                "CCITT Rec. X.731 | ISO/IEC 10164-2": stateChange;;;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21)
managedObjectClass(3) xx2(2)};

generalStringScript MANAGED OBJECT CLASS
DERIVED FROM launchScript;
CHARACTERIZED BY generalStringScriptPackage;;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx3(3)};

asynchronousLaunchPad MANAGED OBJECT CLASS
DERIVED FROM launchPad;
CHARACTERIZED BY triggerAsynchronousResultPackage;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx4(4)};

synchronousLaunchPad MANAGED OBJECT CLASS
DERIVED FROM launchPad;
CHARACTERIZED BY triggerSynchronousResultPackage;
```

```
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx5(5)};

launchPad MANAGED OBJECT CLASS
DERIVED FROM basicSpawnerClass, scriptReferencer;
CHARACTERIZED BY
        launchPadPackage,
          triggerActionAccepter,
          parameterPasser,
          triggerResultPackage,
          triggerEventAccepter,
          terminateAccepter,
          "CCITT Rec. 721 | ISO/IEC 10165-2:1992": externalScheduler,
          suspendResumeAccepter;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx6(6)};

launchScript MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721 | ISO/IEC 10165-2:1992":top;
CHARACTERIZED BY
        launchScriptPackage PACKAGE
        BEHAVIOUR launchScriptBehaviour BEHAVIOUR
        DEFINED AS "This managed object represents instructions to be carried out by a
        command sequencer.";;
        ATTRIBUTES
                scriptId GET,
                executionResultType GET,
                "CCITT Rec. X.721 | ISO /IEC 10165-2:1992": administrativeState
        GETREPLACE;;;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx7(7)};

--
-- The following non-instantiable superclass simplifies the description of the
-- relationship between a launch pad and its scripts, along with the description
-- of the relationship between threads and scripts. Both the launch pad and
-- thread classes include it in their inheritance hierarchies.
--

scriptReferencer MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. X.725 | ISO/IEC 10165-7": genericRelationshipObject;
CHARACTERIZED BY scriptReferencerPackage;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx8(8)};

thread MANAGED OBJECT CLASS
        DERIVED FROM basicSpawnerClass, scriptReferencer;
        CHARACTERIZED BY threadPackage, executionResultPackage;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx9(9)};

suspendableThread MANAGED OBJECT CLASS
        DERIVED FROM thread;
        CHARACTERIZED BY suspendResumeAccepter;
```

```
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3)
xx10(10)};
```

الف-۲ تعاریف بسته

الف-۲-۱ بسته‌های پایه

```
basicSpawnerPackage PACKAGE
        BEHAVIOUR spawnerBehaviour;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx1(1)};

generalStringScriptPackage PACKAGE
        BEHAVIOUR generalStringScriptBehaviour;
        ATTRIBUTES scriptLanguageName GET-REPLACE,
                scriptContent GET-REPLACE;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx2(2)};

parameterPasser PACKAGE
        BEHAVIOUR
        parameterPasserBehaviour;
        ATTRIBUTES
                "CCITT Rec. X.721 | ISO /IEC 10165-2:1992": administrativeState,
                "CCITT Rec. X.721 | ISO /IEC 10165-2:1992": operationalState,
                "CCITT Rec. X.721 | ISO /IEC 10165-2:1992": usageState,
                "CCITT Rec. X.721 | ISO /IEC 10165-2:1992": availabilityStatus;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx3(3)};

executionResultPackage PACKAGE
        BEHAVIOUR executionResultBehaviour;;
        NOTIFICATION executionResultInfo;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx4(4)};

launchPadPackage PACKAGE
        BEHAVIOUR launchPadBehaviour;
        ATTRIBUTES launchPadId GET;
        NOTIFICATIONS
        "CCITT Rec. X.721 | ISO/IEC 10165-2:1992": processingErrorAlarm;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx5(5)};

scriptReferencerPackage PACKAGE
        BEHAVIOUR scriptReferencerBehaviour;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx6(6)};

suspendResumeAccepter PACKAGE
        BEHAVIOUR suspendResumeBehaviour;
        ATTRIBUTES "CCITT Rec. X.721 | ISO/IEC 10165-2:1992":controlStatus GET;
        ACTIONS suspend, resume;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx7(7)};

terminateAccepter PACKAGE
```

```
        ACTIONS terminate;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx8(8)};

threadPackage PACKAGE
        BEHAVIOUR threadBehaviour,
                simpleScriptExecutionBehaviour;
        ATTRIBUTES scriptId GET,
                threadId GET,
                executingParameters GET SET-BY-CREATE,
                        "CCITT Rec. X.731|ISO/IEC 10164-2:1992": operationalState GET;
        NOTIFICATIONS
                "CCITT Rec. X.734|ISO/IEC 10164-5": processingErrorAlarm;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx9(9)};

triggerActionAccepter PACKAGE
        BEHAVIOUR spawnerBehaviour,
                        triggerActionAccepterBehaviour;;
        ATTRIBUTES defaultExecutionParameterList REPLACE WITH DEFAULT
                GET-REPLACE
                SET BY CREATE
                DEFAULT VALUE CSModule.emptyExecutionParameterList;
          availableScriptList REPLACE WITH DEFAULT
                ADD-REMOVE
                GET-REPLACE
                SET BY CREATE
                DEFAULT VALUE CSModule.emptyScriptList;
        ACTIONS Trigger;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx10(10)};

triggerEventAccepter PACKAGE
        BEHAVIOUR triggerEventAccepterBehaviour;
        ATTRIBUTES
                "ITU-T Rec. X.739 (1993)|ISO/IEC 10164-11:1994": observedObjectInstance
        GETREPLACE,
                "ITU-T Rec. X.739 (1993)|ISO/IEC 10164-11:1994": observedAttributeId
        GETREPLACE;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx11(11)};
triggerAsynchronousResultPackage PACKAGE
        BEHAVIOUR triggerAsynchronousResultBehaviour;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx12(12)};

triggerSynchronousResultPackage PACKAGE
        BEHAVIOUR triggerSynchronousResultBehaviour;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx13(13)};

triggerResultPackage PACKAGE
        BEHAVIOUR triggerResultBehaviour;;
        NOTIFICATION triggerResultInfo;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx14(14)};
```

spawnerBehaviour BEHAVIOUR

DEFINED AS !Instances of this class are capable of causing the creation of new object instances. The newly created instances will be contained by this instance, and their names will be automatically generated. Until all created objects are complete, this object's usage status will be "in use". If this object's administrative state is "locked" such new objects cannot be created. If, due to local resource limitations, this object is incapable of supporting more contained objects, its usage status will be "busy".

When an instance of this class causes the creation of new objects, it serves as an IVMO during the creation by supplying values for the new object's attributes based on its own defaultExecutionParameterList attribute or any parameters which were supplied to it as part of the action or local mechanism which triggered the spawning of the new instance.

An instance of this class may cause the creation of new object instances from a single script id, from a set of script ids in any order or from a sequence of script ids in the order specified in the list, i.e. after the first has been created the second may not be created until after the first is completed, and so on. The value of the created object's scriptId attribute gets its value from the corresponding element of this object's script list.!;

executionResultBehaviour BEHAVIOUR
DEFINED AS "Instances of a class supporting this behaviour report intermediate and final results from execution of a thread.";

triggerAsynchronousResultBehaviour BEHAVIOUR

DEFINED AS "As soon as all threads that must be launched by one trigger is launched, the launch pad issues the triggerResultInfo notification.";

triggerSynchronousResultBehaviour BEHAVIOUR

DEFINED AS "As soon as all threads that must be launched by one trigger have completed, the launch pad issues the triggerResultInfo notification which contains execution results or errors.";

triggerResultBehaviour BEHAVIOUR

DEFINED AS "The launch pad issues the triggerResultInfo notification. ";

scriptReferencerBehaviour BEHAVIOUR

DEFINED AS "A script referencer is a non-instantiable object class which defines a reference relationship mapping from instances of the launch pad and the launch script managed object classes and from instances of the thread and the launch

script managed object classes.";

suspendResumeBehaviour BEHAVIOUR
DEFINED AS "Execution of a script by a thread may be suspended by a suspend action directed at the thread or launch pad and subsequently resumed by a resume action_ Default value of controlStatus is empty. If the suspend action is performed, the value changes to suspended. After the resume action is performed, the value changes back to empty.";

triggerEventAccepterBehaviour BEHAVIOUR
DEFINED AS "The launch pad has attributes to monitor a specific attribute in a specific object instance. If the value of the monitored attribute is changed, a trigger to launch the scripts specified by the script ids specified by the default execution parameter list attribute is generated. In the case that the monitored attribute is a counter of EDC (Event Discrimination Counter) defined in Annex C, the notifications through the EDC trigger the launching of script execution by the launch pad.";

triggerActionAccepterBehaviour BEHAVIOUR
DEFINED AS "When an instance of this class which is on duty receives a trigger, from a trigger activator, if its scriptId attribute is not empty, a new object is created in which the script id and class of the new instance come from the value of this instance's scriptId attribute and any of its other attributes and any parameters carried by the trigger.";

threadBehaviour BEHAVIOUR
DEFINED AS "When an instance of an object of this class is created, it begins execution of the command sequence specified through its attributes, using its parameter list to supply any parameters needed by the script. When execution of this sequence is complete, the object is deleted. If execution of the script causes the creation of contained threads, this thread is not considered complete until all contained threads are complete.";

parameterPasserBehaviour BEHAVIOUR
DEFINED AS "An instance of an object of this class passes a set of parameters to an instance of an object of another class.";

simpleScriptExecutionBehaviour BEHAVIOUR
DEFINED AS "A script is executed or interpreted by local means. Its execution status mirrors the following states:
- activated (spontaneous transition to next state: executing);
- executing (next state: timed out or completed);
- timed out (spontaneous transition to next state: completed);
- completed.

NOTE – Timeout value is implementation dependent.";

generalStringScriptBehaviour BEHAVIOUR

DEFINED AS "The syntax and semantics of scripting language which can be represented as general string. See Annexes F and G for details on one such language, SMSL.";


```
availableScriptList ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.AvailableScriptList;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        availableScriptListBehaviour BEHAVIOUR
        DEFINED AS "A set of managed object instance names of the script instructions
        which can be executed by a launch pad.";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx1(1)};

commandSequencerId ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.CommandSequencerId;
        MATCHES FOR EQUALITY;
        BEHAVIOUR

        commandSequencerIdBehaviour BEHAVIOUR
        DEFINED AS "The managed object instance name of the command sequencer.";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx2(2)};

executionResultType ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.ExecutionResultType;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        executionResultTypeBehaviour BEHAVIOUR
        DEFINED AS "This indicates the type of execution result.";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx3(3)};

scriptContent ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.ScriptContent;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        scriptContentBehaviour BEHAVIOUR
        DEFINED AS "The contents of a launch script represented by a general string.";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx4(4)};

scriptId ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.ScriptId;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        scriptIdBehaviour BEHAVIOUR
        DEFINED AS "The managed object instance name of the script to be executed.";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx5(5)};

launchPadId ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.LaunchPadId;
        MATCHES FOR EQUALITY;
```

```
        BEHAVIOUR
        launchPadIdBehaviour BEHAVIOUR
        DEFINED AS "The managed object instance name of the launch pad.";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx6(6)};

scriptLanguageName ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.ScriptLanguageName;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        scriptLanguageNameBehaviour BEHAVIOUR
        DEFINED AS "The managed object instance name of a launch script represented by a
        general string.";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx7(7)};

defaultExecutionParameterList ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.ExecutionParameterList;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        defaultExecutionParameterListBehaviour BEHAVIOUR
        DEFINED AS "A set of managed object instance names of the script instructions and
        parameter values (if required) as inputs to instances to be executed by
        default.";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx8(8)};

executingParameters ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.ExecutionParameter;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        executingParametersBehaviour BEHAVIOUR
        DEFINED AS "A set of managed object instance names of the script instructions and
        parameter values (if required) as inputs to script executions.";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx9(9)};

threadId ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.ThreadId;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        threadIdBehaviour BEHAVIOUR
        DEFINED AS "The managed object instance name of a thread executing script
        instruction(s).";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx10(10)};

triggerId ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.TriggerId;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        triggerIdBehaviour BEHAVIOUR
        DEFINED AS "The managed object instance name of a trigger initiating the execution
        of a launch script.";;
```

REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx11(11)};

الف-۵ تعاریف اعلان

executionResultInfo NOTIFICATION
WITH INFORMATION SYNTAX CSModule.ExecutionResultInfo;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) notification(10) xx1(1)};

triggerResultInfo NOTIFICATION
WITH INFORMATION SYNTAX CSModule.TriggerResultInfo;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) notification(10) xx2(2)};

الف-۶ تعاریف اقدام

resume ACTION
BEHAVIOUR resumeBehaviour BEHAVIOUR
DEFINED AS "An action directed at a basicSpawnerClass object, causing unconditional resumption of all script executions by the basicSpawnerClass object which were initiated by a particular trigger. The value of controlStatus becomes empty as the result of a resume action.";;
WITH INFORMATION SYNTAX CSModule.SpawnerObjectId;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx1(1)};

suspend ACTION
BEHAVIOUR suspendBehaviour BEHAVIOUR
DEFINED AS "An action directed at a basicSpawnerClass object, causing unconditional suspension of all script executions by the basicSpawnerClass object which were initiated by a particular trigger. The value of controlStatus becomes ‘suspended’ as a result of a suspend action.";;
WITH INFORMATION SYNTAX CSModule.SpawnerObjectId;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx2(2)};

terminate ACTION
BEHAVIOUR terminateBehaviour BEHAVIOUR
DEFINED AS "An action directed at a launch pad, causing unconditional termination of all scripts by the launch pad, which was initiated by a particular trigger.";;
WITH INFORMATION SYNTAX CSModule.TriggerId;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx3(3)};

trigger ACTION
BEHAVIOUR triggerBehaviour BEHAVIOUR
DEFINED AS "An initiator of script execution by causing a launch pad to spawn one or more threads.";;
WITH INFORMATION SYNTAX CSModule.TriggerParameters;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx4(4)};

الف-۷ تعاریف انقیاد نام

commandSequencer-system NAME BINDING

```
        SUBORDINATE OBJECT CLASS commandSequencer AND SUBCLASSES;
        NAMED BY SUPERIOR OBJECT CLASS system AND SUBCLASSES;
        WITH ATTRIBUTE commandSequencerId;
        BEHAVIOUR csSystemContainmentBehaviour BEHAVIOUR
        DEFINED AS "Superior object class is system and subordinate object
        class is commandSequencer.";;
        CREATE WITH-REFERENCE-OBJECT, WITH-AUTOMATIC-INSTANCE-NAMING;
        DELETE ONLY-IF-NO-CONTAINED-OBJECTS;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) nameBinding(6) xx1(1)};

launchPad-commandSequencer NAME BINDING
        SUBORDINATE OBJECT CLASS launchPad AND SUBCLASSES;
        NAMED BY SUPERIOR OBJECT CLASS commandSequencer AND SUBCLASSES;
        WITH ATTRIBUTE launchPadId;

        BEHAVIOUR lpCsContainmentBehaviour BEHAVIOUR
        DEFINED AS "Naming a command sequence launch pad_with respect to a command
        sequencer indicates that the command sequencer is the service provider for the
        launch pad.";;
        CREATE WITH-AUTOMATIC-INSTANCE-NAMING;
        DELETE DELETES-CONTAINED-OBJECTS;
        REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21)
        nameBinding(6) xx2(2)};

thread-synchronousLaunchPad NAME BINDING
        SUBORDINATE OBJECT CLASS thread AND SUBCLASSES;
        NAMED BY SUPERIOR OBJECT CLASS synchronousLaunchPad AND SUBCLASSES;
        WITH ATTRIBUTE threadId;
        BEHAVIOUR threadSyncLpContainmentBehaviour BEHAVIOUR
        DEFINED AS "The superior object class synchronousLaunchPad acts as an IVMO for the
        subordinate object class thread.";;
        DELETE DELETES-CONTAINED-OBJECTS;
        REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21)
nameBinding(6) xx3(3)};

        suspendableThread-asynchronousLaunchPad NAME BINDING
        SUBORDINATE OBJECT CLASS suspendableThread AND SUBCLASSES;
        NAMED BY SUPERIOR OBJECT CLASS asynchronousLaunchPad AND SUBCLASSES;
        WITH ATTRIBUTE threadId;
        BEHAVIOUR threadAsyncLpContainmentBehaviour BEHAVIOUR
        DEFINED AS "The superior object class asynchronousLaunchPad acts as an IVMO for
        the subordinate object class suspendable thread.";;
        DELETE DELETES-CONTAINED-OBJECTS;
        REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21)
nameBinding(6) xx4(4)};

thread-thread NAME BINDING
        SUBORDINATE OBJECT CLASS thread AND SUBCLASSES;
        NAMED BY SUPERIOR OBJECT CLASS thread AND SUBCLASSES;
```

```
        WITH ATTRIBUTE threadId;
        BEHAVIOUR threadContainmentBehaviour BEHAVIOUR
        DEFINED AS "The superior object class thread acts as a spawner of the subordinate
        object class thread.";;
        DELETE DELETES-CONTAINED-OBJECTS;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) nameBinding(6) xx5(5)};

suspendableThread-suspendableThread NAME BINDING
        SUBORDINATE OBJECT CLASS suspendableThread AND SUBCLASSES;
        NAMED BY SUPERIOR OBJECT CLASS suspendableThread AND SUBCLASSES;
        WITH ATTRIBUTE threadId;
        BEHAVIOUR suspendableThreadContainmentBehaviour BEHAVIOUR

        DEFINED AS "The superior object class suspendable thread acts as a spawner of the
        subordinate object class suspendable thread.";;
        DELETE DELETES-CONTAINED-OBJECTS;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) nameBinding(6) xx6(6)};

launchScript-system NAME BINDING
        SUBORDINATE OBJECT CLASS launchScript AND SUBCLASSES;
        NAMED BY SUPERIOR OBJECT CLASS system AND SUBCLASSES;
        WITH ATTRIBUTE scriptId;
        BEHAVIOUR lsSystemContainmentBehaviour BEHAVIOUR
        DEFINED AS "The superior object class is system and subordinate object
        class is launchScript.";;
        CREATE WITH-REFERENCE-OBJECT, WITH-AUTOMATIC-INSTANCE-NAMING;
        DELETE ONLY-IF-NO-CONTAINED-OBJECTS;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) nameBinding(6) xx7(7)};
```

الف-۸ تعاریف ASN.1

```
CSModule {joint-iso-itu-t ms(9) function(2) part21(21) asn1Module(2) 0}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN

-- EXPORTS everything

IMPORTS
        SimpleNameType
        FROM Attribute-ASN1Module {joint-iso-itu-t ms(9) smi(3) part2(2)
asn1Module(2) 1 }
        ObjectInstance, Attribute, CMISSync, CMISFilter, ModifyOperator, Scope,
        BaseManagedObjectId FROM CMIP-1 {joint-iso-itu-t ms(9) cmip(1) modules(0)
protocol(3)}
        AE-title FROM ACSE-1 {joint-iso-itu-t association-control(2) abstract-syntax(1)
apdus(0) version(1)};

        cmdSeqRelationshipClasses OBJECT IDENTIFIER ::= {joint-iso-itu-t ms(9) function(2)
```

```
part21(21) relationshipClass(11) }
        cmdSeqRelationshipMappings OBJECT IDENTIFIER ::= {joint-iso-itu-t ms(9) function(2)
part21(21) relationshipMapping(12)}
        cmdSeqRelationshipRoles OBJECT IDENTIFIER ::= {joint-iso-itu-t ms(9) function(2)
part21(21) relationshipRole(13)}

--
-- Range Constraints used for relationship class definitions
--
RangeFromOneToOne ::= INTEGER (1 .. 1)

RangeFromZeroToMax ::= INTEGER (0 .. MAX)

--


-- Counter size constraint
--
MaxCounterSize ::= INTEGER{unlimited(0)}-- size in octets

ExecutionResultInfo ::= SEQUENCE {triggerId TriggerId,
                                  scriptId ScriptId,
                                  threadId ThreadId,
                                  errorCode ErrorCode,
                                  executionResultType ExecutionResultType,
                                  executionResult SET OF Attribute}

TriggerResultInfo ::= SEQUENCE {triggerId TriggerId,
                            CHOICE {singleTriggerResult ResultInfoFromThread,
                                  sequentialTriggerResult SEQUENCE OF
                                              ResultInfoFromThread,
                                  parallelTriggerResult SET OF ResultInfoFromThread}}

ResultInfoFromThread ::= SEQUENCE{executionType ExecutionType,
                                  errorCode ErrorCode,
                                  executionResultType ExecutionResultType,
                                  executionResult SET OF Attribute}

ExecutionType ::= CHOICE {singleExecution ScriptThreadSet,
                            parallelExecution SET OF ScriptThreadSet,
                            sequentialExecution SEQUENCE OF ScriptThreadSet}

ScriptThreadSet ::= SEQUENCE {scriptId ScriptId,
                              hreadId ThreadId}

SpawnerObjectId ::= SEQUENCE {triggerId TriggerId,
                              CHOICE { threadId ThreadId,
                                       launchPadId LaunchPadId}}
```

```
ExecutionResultType ::= OBJECT IDENTIFIER
CommandSequencerId ::= ObjectInstance
ScriptId ::= ObjectInstance
ThreadId ::= ObjectInstance
TriggerId ::= ObjectInstance
LaunchPadId ::= ObjectInstance

ScriptList ::= CHOICE {scriptId ScriptId,
                       sequentialScriptList SEQUENCE OF ScriptId,
                       parallelScriptList_SET OF ScriptId}
AvailableScriptList ::= SET OF ScriptList
emptyScriptList AvailableScriptList ::= {}
emptyExecutionParameterList ExecutionParameterList ::= sequentialExecutionList:{}

TriggerParameters ::= SEQUENCE {triggerId TriggerId,
                                executionParameterList ExecutionParameterList}

ExecutionParameterList ::= CHOICE {executionParameter ExecutionParameter,
                                   sequentialExecutionList SEQUENCE OF
                                   ExecutionParameter,
                                   parallelExecutionList SET OF
                                   ExecutionParameter}

ExecutionParameter ::= SEQUENCE {scriptId ScriptId,
                                 scriptParameters SEQUENCE OF Attribute}

emptyParameterList ExecutionParameterList ::= sequentialExecutionList:{ }

ErrorCode ::= SET OF INTEGER {noError(0),
                              noScriptError(1),
                              scriptRejectedError(2),
                              invalidParameterTypeError(3),
                              invalidParameterValueError(4),
                              scriptSyntaxError(5),
                              scriptExecutionFailedError(6),
                              invalidParmeterNumber(7),
                              unauthorizedAccessError(8)}

ScriptLanguageName ::= OBJECT IDENTIFIER
ScriptContent ::= GeneralString
ModificationList ::= SET OF SEQUENCE{modifyOperator [2] IMPLICIT
                                     ModifyOperator DEFAULT replace,
                                     attributeId AttributeId,
                                     attributeValue ANY DEFINED BY
                                     attributeId OPTIONAL
                                     -- absent for setToDefault
                                     }
END
```

پیوست ب

(الزامی)

مدل رابطه عمومی

GRM برای ترتیب‌سنج فرمان به شرح زیر است:

```
--
-- The following relationship classes support the command sequencer model
--
commandSequencer-launchPadRelationshipClassBehaviour
BEHAVIOUR DEFINED AS
        !
                The relationship class is concerned with the relationship between a command
                sequencer and its launch pads used to initiate the execution of scripts by
                means of threads requiring the support services provided by the command
                sequencer.
        !;

commandSequencer-LaunchPad-bindingBehaviour
BEHAVIOUR DEFINED AS
        !
                This notification occurs upon the binding of a launch pad into the command
                sequencer / launch pad relationship.
        !;

commandSequencer-LaunchPad-unbindingBehaviour
BEHAVIOUR DEFINED AS
        !
                This notification occurs when a launch pad is removed from the command
                sequencer / launch pad relationship. A launch pad may be removed from the
                relationship only if its administrative state is locked.
        !;

commandSequencer-launchPad-RelationshipClass
RELATIONSHIP CLASS
        BEHAVIOUR commandSequencer-launchPadRelationshipClassBehaviour,
                commandSequencer-LaunchPad-bindingBehaviour,
           commandSequencer-LaunchPad-unbindingBehaviour;
        SUPPORTS
                ESTABLISH,
                QUERY,
                TERMINATE,
                NOTIFY commandSequencer-LaunchPad-binding,
                NOTIFY commandSequencer-LaunchPad-unbinding;
        ROLE commandSequencerRole
```

```
            COMPATIBLE-WITH commandSequencer
            PERMITTED-ROLE-CARDINALITY                                  CONSTRAINT
        CASN1.RangeFromOneToOne
            REQUIRED-ROLE-CARDINALITY-CONSTRAINT
        CASN1.RangeFromOneToOne

            PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT
            CASN1.RangeFromOneToOne
            REGISTERED AS { CSModule.cmdSeqRelationshipRoles 1 }

        ROLE launchPadRole
            COMPATIBLE-WITH launchPad
            PERMITTED-ROLE-CARDINALITY-CONSTRAINT
        CSModule.RangeFromZeroToMax
            REQUIRED-ROLE-CARDINALITY-CONSTRAINT
        CSModule.RangeFromZeroToMax
            BIND-SUPPORT
            UNBIND-SUPPORT
            PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT
        CSModule.RangeFromOneToOne
        REGISTERED AS { CSModule.cmdSeqRelationshipRoles 2 };
REGISTERED AS { CSModule.cmdSeqRelationshipClasses 1 };

commandSequencer-LaunchPad-RelationshipMapping-Behaviour
BEHAVIOUR DEFINED AS
        !
            This relationship mapping describes how the command sequencer to launch pad
            relationship class may be represented using containment. In this
            relationship mapping, the command sequencer is the superior object for the
            purposes of naming, and launch pads are contained by it. Participation in
            this relationship implies that the launch pad and its spawn may use the
            services provided by the resource represented by the command sequencer.
        !;
commandSequencer-launchPad-RelationshipMapping
RELATIONSHIP MAPPING
        RELATIONSHIP CLASS
        commandSequencer-launchPad-RelationshipClass;
        BEHAVIOUR commandSequencer-launchPad-RelationshipMapping-Behaviour;

        ROLE commandSequencerRole
            RELATED-CLASSES commandSequencer
            REPRESENTED BY NAMING
                launchPad-commandSequencer-NameBinding
                USING SUPERIOR,

        ROLE launchPadRole
            RELATED-CLASSES launchPad
            REPRESENTED BY NAMING
                launchPad-commandSequencer-NameBinding
```

```
                    USING SUBORDINATE;

        OPERATIONS MAPPING
                ESTABLISH MAPS-TO-OPERATION
                        CREATE commandSequencer OF commandSequencerRole
                TERMINATE MAPS-TO-OPERATION
                        DELETE commandSequencer OF commandSequencerRole

                NOTIFY commandSequencer-launchPad-binding
                                MAPS-TO-OPERATION
                                NOTIFICATION "CCITT Rec. X.721 | ISO/IEC 10165-2:1992":
                                objectCreationNotification
                                        OF commandSequencerRole
                NOTIFY commandSequencer-launchPad-unbinding
                                MAPS-TO-OPERATION
                                NOTIFICATION "CCITT Rec. X.721 | ISO/IEC 10165-2:1992":
                                objectDeletionNotification
                                        OF commandSequencerRole
                BIND MAPS-TO-OPERATION
                                CREATE launchPad OF launchPadRole
                UNBIND MAPS-TO-OPERATION
                                DELETE launchPad
                                        OF launchPadRole
                QUERY MAPS-TO-OPERATION
                                GET OF commandSequencerRole
                                GET OF launchPadRole;
REGISTERED AS {CSModule.cmdSeqRelationshipMappings 1}

scriptReferenceRelationshipClassBehaviour
BEHAVIOUR DEFINED AS
        !
                This relationship class describes the relationship existing between scripts
                and an object which references them for the purposes of identifying task(s)
                to be carried out.
        !;

scriptReferencerRelationshipClass
RELATIONSHIP CLASS
        BEHAVIOUR scriptReferencerRelationshipClassBehaviour;
        SUPPORTS
                ESTABLISH,
                TERMINATE,
                QUERY;

        ROLE scriptUserRole
                COMPATIBLE-WITH scriptReferencer
                PERMITTED-ROLE-CARDINALITY-CONSTRAINT
        CSModule.RangeFromOneToOne
```

```
            REQUIRED-ROLE-CARDINALITY                                   CONSTRAINT
        CSModule.RangeFromOneToOne
            PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT
                CSModule.RangeFromZeroToMax
        REGISTERED AS {CSModule.cmdSeqRelationshipRoles 3}

        ROLE scriptRole
            COMPATIBLE-WITH launchScript
            PERMITTED-ROLE-CARDINALITY-CONSTRAINT
        CSModule.RangeFromOneToOne
            REQUIRED-ROLE-CARDINALITY-CONSTRAINT
        CSModule.RangeFromOneToOne

            PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT
                CSModule.RangeFromZeroToMax
        REGISTERED AS {CSModule.cmdSeqRelationshipRoles 4}

REGISTERED AS {CSModule.cmdSeqRelationshipClasses 2}

launchPad-launchScriptRelationshipMappingBehaviour BEHAVIOUR
DEFINED AS
!
        This relationship mapping describes the relationship between the launch pad and the launch
        script. The launch pad initiates execution of the launch script and references it for the
        purposes of execution.
!;

launchPad-LaunchScriptMapping
RELATIONSHIP MAPPING
        RELATIONSHIP CLASS scriptReferenceRelationshipClass;
        BEHAVIOUR launchPad-launchScriptMappingBehaviour;
        ROLE scriptUserRole
           RELATED CLASSES launchPad
        REPRESENTED BY ATTRIBUTE scriptList
        ROLE scriptRole
           RELATED CLASSES launchScript;
        OPERATIONS MAPPING
           ESTABLISH MAPS-TO-OPERATION
                CREATE launchPad OF scriptUserRole
                -- using SET-BY-CREATE of scriptList --
                REPLACE scriptList of scriptUserRole
                -- which effectively adds a scriptId to scriptList --
                ADD scriptList of scriptUserRole,
           TERMINATE MAPS-TO-OPERATION
                DELETE launchPad OF scriptUserRole
                REPLACE scriptList OF scriptUserRole
                -- which effectively removes a scriptId from scriptList --
                REMOVE scriptList OF scriptUserRole,
           QUERY MAPS-TO-OPERATION
```

```
                GET scriptList OF scriptUserRole;
REGISTERED AS {CSModule.cmdSeqRelationshipMappings 2};

thread-launchScriptRelationshipMappingBehaviour BEHAVIOUR
DEFINED AS
!
        This relationship class describes the relationship between a thread and launch
        script. The launch script is referenced by the thread by means of the scriptId
        attribute from the scriptIds in the scriptList.
!;

thread-launchScriptMapping
RELATIONSHIP MAPPING

        RELATIONSHIP CLASS scriptReferenceRelationshipClass;
        BEHAVIOUR thread-launchScriptRelationshipMappingBehaviour;
        ROLE scriptUserRole
          RELATED CLASSES thread
          REPRESENTED BY ATTRIBUTE scriptId
          QUALIFIED BY scriptList
        ROLE scriptRole
          RELATED CLASSES launchScript;
        OPERATIONS MAPPING
          ESTABLISH MAPS-TO-OPERATION
                CREATE OF scriptUserRole,
          TERMINATE MAPS-TO-OPERATION
                DELETE OF scriptUserRole,
          QUERY MAPS-TO-OPERATION
                GET scriptId OF scriptUserRole;
REGISTERED AS {CSModule.cmdSeqRelationshipMappings 3};

spawner-progeny-RelationshipClass
RELATIONSHIP CLASS
        BEHAVIOUR spawner-progenyRelationshipClassBehaviour BEHAVIOUR
        DEFINED AS
                !
                When an instance of this class causes the creation of new objects, it serves as an
                IVMO during the creation by supplying values for the new object's attributes based
                on its own defaultExecutionParameterList attribute and any parameters which were
                supplied to it as part of the action or local Instances of this class are capable of
                causing the creation of new object instances. The newly created instances will be
                contained by this instance, and their names will be automatically generated. Until all
                created objects are complete, this object's usage status will be "in use". If this
                object's administrative state is "locked" such new objects cannot be created. If, due
                to local resource limitations, this object is incapable of supporting more contained
                objects, its usage status will be "busy".

                An instance of this class may cause the creation of new object instances
                from a single scriptId, from a set of scriptIds in any order or from a
```

```
            sequence of scriptIds in the order specified in the list i.e. after the
            first has been created the second may not be created until after the first
            is completed, and so on. The value of the created object's scriptId
            attribute gets its value from the corresponding element of this object's
            scriptList.
            !;
        SUPPORTS
            ESTABLISH
            TERMINATE;

        ROLE spawnerRole COMPATIBLE-WITH basicSpawnerClass
            PERMITTED-ROLE-CARDINALITY-CONSTRAINT
        CSModule.RangeOneToOne
            REQUIRED-ROLE-CARDINALITY-CONSTRAINT CSModule.RangeOneToOne

            PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT
                CSModule.RangeOneToOne
            REGISTERED AS {CSModule.cmdSeqRoles 5}
          ROLE progenyRole COMPATIBLE-WITH thread
            PERMITTED-ROLE-CARDINALITY-CONSTRAINT
        CSModule.RangeZeroToMax
            REQUIRED-ROLE-CARDINALITY-CONSTRAINT CSModule.RangeZeroToMax
            PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT
          CSModule.RangeOneToOne

            REGISTERED AS {CSModule.cmdSeqRoles 6}
REGISTERED AS {CSModule.cmdSeqRelationshipClasses 3};
```

پیوست پ

(الزامی)

تعاریف اطلاعات مدیریت برای شمارش تمایز رویداد

شیء کلاس شمارنده تمایز رویداد (EDC)، تعداد رویدادهای ورودی را می‌شمارد. قبل از شمارش، EDC مقادیر صفات مربوط به رویداد یا شیء ایجادکننده رویداد را آزمایش کرده و آن‌را تمیز می‌دهد.

پ-۱ کلاس شیء مدیریت‌شده

```
eventDiscriminationCounter MANAGED OBJECT CLASS
DERIVED FROM "DMI":discriminator;
CHARACTERIZED BY
edcPackage PACKAGE
        BEHAVIOUR edcBehaviour BEHAVIOUR
        DEFINED AS
        "If the result of discrimination of a potential event report evaluates to TRUE and
        the event discrimination counter is in the Unlocked and Enabled state and does not
        exhibit the off-duty availability status, then the counter value of the counter
        attribute is incremented.";;
                ATTRIBUTES
                        "CCITT Rec. 721 | ISO/IEC 10165-2:1992":counter GET,
                        maxCounterSize GET;
                NOTIFICATIONS
                        "CCITT Rec. 721 | ISO/IEC10162:1992":processingErrorAlarm;;;
CONDITIONAL PACKAGES
        counterAlarmPackage PRESENT IF "a counter is of finite size and a notification is
        triggered by a capacity alarm threshold.";
REGISTERED AS {joint-iso-itu-t ms(9) ms(9) function(2) part21(21)
managedObjectClass(3) xx11(11)};
```

پ-۲ بسته

```
counterAlarmPackage PACKAGE
        BEHAVIOUR
        counterAlarmBehaviour BEHAVIOUR
        DEFINED AS "When the counter value reaches the capacity alarm threshold(as a
                percentage of maximum counter size), the EDC(Event Discrimination Counter)
                generates an event indicating that a capacity threshold has been reached
                or exceeded. In reporting the capacity threshold event, use is made of the
                alarm report defined in CCITT Rec. X.733 | ISO/IEC 10164-4. Only the
                following parameters of the alarm report shall be used and all parameters
                are mandatory when used for reporting counter capacity threshold alarms.
                Managed Object Class - This parameter shall identify the
                counter class.
```

Managed Object Instance - This parameter shall identify the
instance of the counter that generated the event.
Alarm Type - This parameter shall indicate that a processing
error alarm has occurred.

Event time - This parameter carries the time at
which the capacity threshold event occurred.
Perceived Severity - This_parameter will indicate the severity
assigned to the capacity threshold event. When the 100%
counter full condition is reached, a severity value of critical
shall be assigned to this event.
Monitored Attributes - This parameter shall carry the maximum counter size
attribute of the EDC.
Probable Cause - This parameter shall carry the value congestion.
Threshold Info - This parameter shall carry the capacity threshold
value (as percentage of total capacity) that was reached or exceeded in
generating this event.";;
ATTRIBUTES
"CCITT Rec. 721| ISO/IEC 10165-2:1992":capacityAlarmThreshold GET-REPLACE
ADD-REMOVE;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx15(15)};

پ-۳ صفت

maxCounterSize ATTRIBUTE
WITH ATTRIBUTE SYNTAX CSModule.MaxCounterSize;
MATCHES FOR EQUALITY, ORDERING;
BEHAVIOUR
maxSizeOrderingBehaviour BEHAVIOUR
DEFINED AS "This Attribute represent the largest value of the counter. The
ordering in the same as for sequentially increasing positive integers except that
a value of zero is largest and denotes infinite size.";;

REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx12(12)};

پیوست ت

(الزامی)

کلاس شیء پشتیبانی مدیریت cmisScript

یک کلاس شیء مدیریت‌شده اسکریپت، به نام اسکریپت CMIS که می‌تواند برای ساماندهی فراخوانی عملکردهای CMIS تعریف شود، در ادامه آمده است.

ت-۱ صفات

ت-۱-۱ صفات واردشده از تعریف اطلاعات مدیریت

این استاندارد ، به صفات مدیریت زیر که نحو انتزاعی آن‌ها در CCITT Rec. X.721 | ISO/ IEC 10165-2 مشخص شده است، ارجاع می‌کند:

الف) فهرست شناسه صفت؛

ب) کلاس شیء؛

پ) فهرست صفات.

این استاندارد ، به صفات مدیریت زیر که نحو انتزاعی آن‌ها در ITU Rec. X.711 | ISO/ IEC 9596-1 مشخص شده است، ارجاع می‌کند:

الف) همگام‌سازی؛

ب) دامنه کاربرد؛

پ) فیلتر؛

ت) شناسه شیء مدیریت‌شده پایه؛

ث) فهرست تغییرات.

ت-۲ تعاریف

ت-۲-۱ cmisScript

یک شیء پشتیبانی مدیریت، که فراخوانی یک عملکرد CMIS واحد را هدایت می‌کند. پنج نوع اسکریپت CMIS مشخص شده است:

ت-۲-۲ getCmisScript

اسکریپت CMIS که یک عملکرد GET را نشان می‌دهد.

ت-۲-۳ setCmisScript

اسکریپت CMIS که یک عملکرد SET را نشان می‌دهد.

ت-۲-۴ actionCmisScript

اسکریپت CMIS که یک عملکرد ACTION را نشان می‌دهد.

ت-۲-۵ createCmisScript

اسکریپت CMIS که یک عملکرد CREATE را نشان می‌دهد.

ت-۲-۶ deleteCmisScript

اسکریپت CMIS که یک عملکرد DELETE را نشان می‌دهد.

اسکریپت‌های CMIS اسکریپت‌های تک دستورالعملی هستند که جزئیات عملیاتی را به نخ‌ها می‌دهند. به‌عنوان مثال، زبان‌های اسکریپت ممکن است برای ایجاد درخواست CMIS به یک عامل، به سوابق فرمان مناسب رجوع کنند.

ت-۳ getCmisScript

ت-۳-۱ مشخصات getCmisScript

صفات زیر در getCmisScript تعریف شده است:

- شناسه شیء مدیریت‌شده پایه؛
- همگام‌سازی.
- دامنه کاربرد؛
- فیلتر؛
- فهرست شناسه صفت.

ت-۳-۲ بسته‌های getCmisScript

getCmisScript دارای بسته اجباری زیر است:

- getCmisScriptPackage.

ت-۴ setCmisScript

ت-۴-۱ مشخصات setCmisScript

setCmisScript دارای تعاریف صفات زیر است:

- شناسه شیء مدیریت‌شده پایه؛
- همگام‌سازی؛
- دامنه کاربرد؛
- فیلتر؛
- فهرست تغییر.

ت-۴-۲ بسته‌های setCmisScripts

setCmisScript دارای بسته اجباری زیر است:

- setCmisScriptPackage.

ت-۵ actionCmisScript

ت-۵-۱ مشخصات actionCmisScript

actionCmisScript دارای تعاریف صفات زیر است:

- شناسه شیء مدیریت‌شده پایه؛
- همگام‌سازی؛
- دامنه کاربرد؛
- فیلتر؛

ت-۵-۲ بسته‌های actionCmisScript

کلاس ثبت فرمان اقدام دارای بسته الزامی زیر است:

- actionCmisScriptPackage.

ت-۶ createCmisScript

ت-۶-۱ مشخصات createCmisScript

ثبت فرمان اقدام دارای تعاریف صفات زیر است:

- کلاس شیء؛
- فهرست صفت.

ت-۶-۲ بسته‌های eCmisScriptcreat

کلاس ثبت فرمان ایجاد دارای بسته الزامی زیر است:

- createCmisScriptPackage.

و دارای بسته‌های مشروط زیر است:

- managedObjectInstancePackage؛
- superiorObjectInstancePackage؛
- referenceObjectInstancePackage.

ت-۷ deleteCmisScript

ت-۷- ۱ مشخصات deleeteCmisScriptdel

deleteCmisScript دارای تعاریف ویژگی زیر است:

- شناسه شیء مدیریت‌شده پایه؛
- همگام‌سازی؛

- دامنه کاربرد؛
- فیلتر.

ت-۷- ۲ بسته‌های deleteCmisScript

deleteCmisScript دارای بسته الزامی زیر است:

- deleteCmisScriptPackage .

ت-۸ خدمات

ت-۸-۱ تعریف خدمت گزارش دریافت[1]

این بند خدمت گزارش دریافت را مشخص و آن را بر روی خدمات CMIS M-EVENT-REPORT نگاشت می‌کند.

ت-۸-۲ تعریف خدمت گزارش تنظیم[2]

این بند، خدمت گزارش تنظیم را مشخص و آن را بر روی خدمات CMIS M-EVENT-REPORT نگاشت می‌کند.

ت-۸-۳ تعریف خدمت گزارش اقدام[3]

این بند خدمت گزارش اقدام را مشخص و آن را بر روی خدمات CMIS M-EVENT-REPORT نگاشت می‌کند.

ت-۸-۴ تعریف خدمت گزارش ایجاد[4]

این خدمت اجازه می‌دهد تا یک کاربر MIS، در نقش عامل، ایجاد یک شیء مدیریت‌شده را گزارش دهد. این خدمت هم به‌عنوان یک خدمت تأییدشده و هم به‌عنوان یک خدمت تأییدنشده تعریف شده و بر روی خدمات CMIS M-EVENT-REPORT نگاشت می‌شود. این خدمت در CCITT Rec.X. 730 | ISO/ IEC 10164-1 تعریف شده است.

ت-۸-۵ تعریف خدمت گزارش حذف[5]

این خدمت اجازه می‌دهد تا یک کاربر MIS، در نقش عامل، حذف یک شیء مدیریت‌شده را گزارش دهد. این خدمت هم به‌عنوان یک خدمت تاییدشده و هم به‌عنوان یک خدمت تاییدنشده تعریف شده و بر روی خدمات CMIS M-EVENT-REPORT نگاشت می‌شود. این خدمت در CCITT Rec.X. 730 | ISO/ IEC 10164-1 تعریف شده است.

1- Get
2 -Set
3 -Action
4 -Creation
5 -Delete

ت-۹-۱ تعاریف شیء مدیریت‌شده

```
cmisScript MANAGED OBJECT CLASS
	DERIVED FROM launchScript;
CHARACTERIZED BY cmisScriptPackage PACKAGE
	BEHAVIOUR
		cmisScriptBehaviour BEHAVIOUR
	DEFINED AS
	!
		An instance of this managed object class models information necessary to
		execute a single CMIS operation.
	!;;;;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3)
xx12(12)};
getCmisScript MANAGED OBJECT CLASS
	DERIVED FROM cmisScript;
CHARACTERIZED BY getCmisScriptPackage PACKAGE
	BEHAVIOUR

	getCmisScriptBehaviour BEHAVIOUR
	DEFINED AS
	!
		An instance of this managed object class models information
		necessary to execute a single CMIS GET operation.
	!;;
	ATTRIBUTES
		baseManagedObjectId GET,
		synchronization GET-REPLACE,
		scopeGET-REPLACE,
		filter GET-REPLACE,
		"CCITT Rec. X.721 | ISO/IEC 10165-2:1992":attributeIdentifierList
		GET-REPLACE ADD-REMOVE;;;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3)
xx13(13)};

setCmisScript MANAGED OBJECT CLASS
	DERIVED FROM cmisScript;
CHARACTERIZED BY setCmisScriptPackage PACKAGE
	BEHAVIOUR
	setCmisScriptBehaviour BEHAVIOUR
	DEFINED AS
	!
		An instance of this managed object class models information necessary to
		execute a single CMIS SET operation.
	!;;
	ATTRIBUTES
		baseManagedObjectId GET,
		synchronization GET-REPLACE,
```

```
                scope GET-REPLACE,
                filter GET-REPLACE,
                modificationList GET-REPLACE ADD-REMOVE;;;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3)
xx14(14)};

actionCmisScript MANAGED OBJECT CLASS
        DERIVED FROM cmisScript;
        CHARACTERIZED BY
actionCmisScriptPackage PACKAGE BEHAVIOUR
                actionCmisScriptBehaviour BEHAVIOUR
                DEFINED AS
                !
                An instance of this managed object class models information necessary to
                execute a single CMIS ACTION operation.
                !;;

                ATTRIBUTES
                        baseManagedObjectId GET,
                        synchronization GET-REPLACE,
                        scope GET-REPLACE,
                        filter GET-REPLACE;;;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3)
xx15(15)};

createCmisScript MANAGED OBJECT CLASS
DERIVED FROM cmisScript;
CHARACTERIZED BY
        createCmisScriptPackage PACKAGE
        BEHAVIOUR
                createCmisScriptBehaviour BEHAVIOUR
                DEFINED AS
                !
                        An instance of this managed object class models information necessary
                        to execute a single CMIS CREATE operation.
                !;;
        ATTRIBUTES
                "CCITT Rec. 721 | ISO/IEC 10165-2:1992": objectClass GET,
                "CCITT Rec. 721 | ISO/IEC 10165-2:1992": attributeList GET-REPLACE
                ADDREMOVE;;;
        CONDITIONAL PACKAGES
                managedObjectInstancePackage
                        PRESENT IF "the superiorObjectInstancePackage is not present.",
                superiorObjectInstancePackage
                        PRESENT IF "the managedObjectInstance Package is not present.",
                referenceObjectInstancePackage
                        PRESENT IF "the manager has the specified value.";
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3)
xx16(16)};
```

```
deleteCmisScript MANAGED OBJECT CLASS
DERIVED FROM cmisScript;
CHARACTERIZED BY
        deleteCmisScriptPackage PACKAGE
        BEHAVIOUR
        deleteCmisScriptBehaviour BEHAVIOUR
        DEFINED AS
        !
                An instance of this managed object class models information necessary to
                execute a single CMIS DELETE operation.
        !;;
        ATTRIBUTES
                baseManagedObjectId GET,
                synchronization GET-REPLACE,
                scope GET-REPLACE,
                filter GET-REPLACE;;;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3)
xx17(17)};
```

ت-۹-۲ تعاریف بسته

```
_managedObjectInstancePackage PACKAGE
ATTRIBUTES
        managedObjectInstance GET-REPLACE;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx16(16)};

superiorObjectInstancePackage PACKAGE
        ATTRIBUTES
        superiorObjectInstance GET-REPLACE;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx17(17)};

        referenceObjectInstancePackage PACKAGE
        ATTRIBUTES
        referenceObjectInstance GET-REPLACE;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx18(18)};
```

ت-۹-۳ تعاریف صفت

```
baseManagedObjectId ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.BaseManagedObjectId;
        MATCHES FOR EQUALITY;
        BEHAVIOUR baseManagedObjectIdBehaviour
        BEHAVIOUR
        DEFINED AS
        !
                This is the identifier for the information on the CMIS
                operation to be executed.
        !;;
        REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx13(13)
};
```

```
scope ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.Scope;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        scopeBehaviour BEHAVIOUR
        DEFINED AS
        !
                This is the first phase in the selection of managed object(s) to which the
                CMIS script operations should be directed. It indicates the managed
                object(s) to which a filter should be applied.
        !;;
         REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7)
xx14(14)};

filter ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.CMISFilter;
        MATCHES FOR EQUALITY;


        BEHAVIOUR
        filterBehaviour BEHAVIOUR
        DEFINED AS
        !
                This is the second phase in the selection of managed object(s) to which
                the CMIS script operations should be directed. A set of tests is applied to
                each of the previously scoped managed objects to extract a subset.
        !;;
        REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx15(15)
};

synchronization ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.CMISSync;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        synchronizationBehaviour BEHAVIOUR
        DEFINED AS
        !
                This indicates the manner in which operations are to be synchronized across
                managed object instances when multiple managed objects have been selected by the
                scoping and filtering mechanisms.
        !;;
        REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx16(16)
};

modificationList ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.ModificationList;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
```

```
        modificationListBehaviour BEHAVIOUR
        DEFINED AS
        !
                This represents the list of attributes to be modified by the CMISSet script
                and contains the values to which these attributes should be set.";;
        !;;
        REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx17(17)
};

superiorObjectInstance ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.ObjectInstance;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        superiorObjectInstanceBehaviour BEHAVIOUR
        DEFINED AS
        !
                This attribute identifies the existing managed object instance
                which is supplied, the managedObjectInstance attribute shall not
                be supplied.
        !;;

REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx18(18)};

referenceObjectInstance ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.ObjectInstance;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        referenceObjectInstanceBehaviour BEHAVIOUR
        DEFINED AS
        !
                The managed object instance name of the same class as the managed object to be
                created. Attribute values associated with the reference object instance become
                default values for those not specified by the attribute list attribute.
        !;;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx19(19)};
```

پیوست ث

(اطلاعاتی)

کلاس شیء مدیریت‌شده CMIP_CS

در ادامه، مثالی آمده است که نشان می‌دهد چگونه می‌توان زیرکلاس‌های اضافی‌ای تعریف کرد که حاوی صفاتی مانند عنوان AE باشند.

ث-۱ cmipCS

ث-۱-۱ مرور کلی

- زیرکلاس ماشین پروتکل cmip و ترتیب‌سنج فرمان.
- عنوان AE فراخواننده را برای ترتیب‌سنج فرمان تعریف می‌کند.

ث-۱-۲ مشخصات کلاس شیء مدیریت شده cmipCS

- aetitle.

ث-۱-۳ بسته‌های cmipCS

کلاس شیء مدیریت شده به‌وسیله cmipCS دارای بسته‌ی الزامی‌زیر است:

- بسته aetitle.

ث-۱-۴ تعاریف GDMO

```
cmipCS MANAGED OBJECT CLASS
DERIVED FROM commandSequencer;
CHARACTERIZED BY aeTitlePackage;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3)
xx18(18)};

aeTitlePackage PACKAGE
ATTRIBUTES
        aetitle GET;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx19(19)};
```

ث-۱-۵ تعاریف صفت

```
aetitle ATTRIBUTE
        WITH ATTRIBUTE SYNTAX CSModule.AE-title;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        aeTitleBehaviour BEHAVIOUR
        DEFINED AS "An instance of this managed object class defines the
        calling AE title for the command sequencer";;
REGISTERED AS {joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx20(20)};
```

پیوست ج

(الزامی)

زبان اسکریپت نویسی مدیریت سامانه‌ها (SMSL[1])

این پیوست یک زبان اسکریپت‌نویسی رشته‌ای عمومی، به نام SMSL را به‌منظور نوشتن رویه‌هایی برای اجرا در یک محیط ترتیب‌سنج فرمان تعریف می‌کند.

SMSL به‌همراه توابع مورد نیاز برای این محیط ارائه شده است. برای انجام این کار، SMSL برخی از ویژگی‌های کامل بودن زبان‌هایی مثل C، csh، Perl، یا awk را قربانی می‌کند، در حالی که برخی از دستورات و توابعی را که آن زبان‌ها را بسیار قدرتمند و محبوب کرده‌اند، پیاده‌سازی می‌کند.

ج-۱- نگاشت GDMO به SMSL

SMSL دارای یک ماشین مجازی است که قادر به تفسیر GDMO است (به شکلی که در سری استانداردهای ISO/IEC 10165 و 10164 تعریف شده است) که به انواع داده SMSL مطابق با مدل شیء SMSL (به شکلی که در زیربند ج-۱۴-۱۱ توصیف شده) نگاشت شده است. اسکریپت‌های نوشته شده با SMSL به وسیله ماشین مجازی SMSL تفسیر شده و اجرا می‌شوند. در محیط ترتیب‌سنج فرمان، ماشین مجازی SMSL در نقش مدیر برای اهداف مدیریت سامانه‌ها عمل می‌کند. ماشین مجازی SMSL مجازی باید بتواند همه انواع داده SMSL را که به‌وسیله اسکریپت‌های SMSL استفاده می‌شوند، تفسیر کند ، یعنی از دیدگاه نویسنده اسکریپت، تمام انواعی که در اسکریپت استفاده شده‌اند، بدون در اختیار داشتن تعاریف این انواع به‌طور ضمنی برای ماشین مجازی شناخته شده باشند. ماشین مجازی SMSL باید بتواند همه نمونه‌های SMSL را که در حال حاضر نمونه‌سازی شده‌اند، تفسیر کند ، یعنی از دیدگاه نویسنده اسکریپت، تمام ارجاعات به اشیائی که از قبل نمونه‌سازی شده‌اند، برای ماشین مجازی شناخته شده باشد.

ج-۲- توابع داخلی[2] SMSL

SMSL حاوی تعدادی از توابع داخلی برای ایجاد و دستکاری اشیاء و توابع همه منظوره همچون توابع ریاضی، منطقی، یا ورودی/خروجی است. در ادامه خلاصه‌ای از توابع داخلی SMSL آمده است. توابع، به‌صورت جداگانه در زیر توضیح داده شده‌اند.

ج-۲-۱ توابعی برای کنترل همزمانی

SMSL حاوی دو تابع داخلی برای اجرای کنترل همزمانی است: ()lock و ()unlock. معمولاً این توابع برای خطی کردن (به‌صورت طولی در آوردن) دسترسی‌ها به ساختار داده‌ها و منابع مشترک استفاده می‌شوند.

تمام فرآیندهای SMSL که سعی در خطی کردن دسترسی‌ها به یک منبع دارند، باید به‌وسیله‌ی درخواست قفل کردن بر روی یک نام قفل داده شده، با یکدیگر همکاری کنند. همه دسترسی‌ها به منبع، از جمله توابع ()set و

---

1 -System Management Scripting Language
2 -Built-in

()get، بدون استفاده از یک قفل برای دسترسی به منابع مشترک، رد می‌شوند. این مسئله به عهده هر فرآیند SMSL است که فقط زمانی به یک منبع دسترسی پیدا کند که قفل مورد نیاز را در اختیار داشته باشد.

ج-۳ توابع Set برای فهرست‌های SMSL

زبان SMSL حاوی توابع زیر برای انجام عملیات set بر روی فهرست‌های SMSL است؛

- ()difference برای بازگرداندن فهرست عناصر مختلف بین فهرست‌ها؛
- ()intersection برای بازگرداندن یک فهرست از عناصر مشترک بین فهرست‌ها؛
- ()sort برای بازگرداندن یک فهرست به ترتیب صعودی یا نزولی عنصر؛
- ()subset برای بررسی اینکه آیا یک فهرست درون دیگری قرار دارد؛
- ()union برای بازگرداندن فهرستی که ترکیبی از فهرست‌ها است؛
- ()unique برای بازگرداندن فهرستی از عناصر که تنها در یک فهرست حضور دارند.

این توابع، فهرست‌های SMSL را به‌عنوان مجموعه‌ای از عناصر پردازش می‌کنند. هر عضو از یک فهرست، رشته متنی است که با یک کاراکتر جای خالی خاتمه پیدا می‌کند.

مجموعه NULL [""] معادل مجموعه تهی یا خالی (ϕ) در نظریه مجموعه‌ها است. مجموعه NULL به‌وسیله توابع مجموعه‌ای SMSL به‌عنوان یک مجموعه مناسب[1] که حاوی هیچ عنصری نیست، در نظر گرفته می‌شود. رشته NULL، یا []، یک عنصر فهرست SMSL است که دارای هیچ کاراکتری نیست. توابع مجموعه‌ای SMSL اجازه می‌دهند که فهرست‌ها حاوی رشته NULL باشند.

مفهوم مورد نظر SMSL از یک مجموعه، فهرست منحصربه‌فردی از عناصر صعودی یا نزولی که مفهومی آشنا در نظریه مجموعه‌ها است، نیست. در بسیاری از موارد، فهرست‌های LSMS حاوی عناصر تکراری است که بدون هیچ ترتیب خاصی قرار گرفته‌اند. یک فهرست SMSL می‌تواند با استفاده از تابع ()unique برای حذف تکرارها، و تابع ()sort برای مرتب سازی عناصر به ترتیب صعودی یا نزولی، به یک مجموعه مرتب تبدیل شود.

ج -۴ توابع ریاضی SMSL

SMSL از یک زیرمجموعه پایه از توابع ریاضی پشتیبانی می‌کند. این توابع همگی شبیه به توابع ریاضی زبان C هستند.

توابع ریاضی SMSL شامل بعضی از وارسی‌های خطای زمان اجرا برای محدوده و دامنه هستند. هر دو وضعیت باعث ایجاد یک پیغام خطای زمان اجرا می‌شوند که متغیر شماره خطای SMSL را به یک مقدار مناسب تنظیم می‌کند.

علاوه بر این، هر مقدار غیرعددی تولیدشده به‌وسیله چاپ نتیجه فراخوانی تابع، مانند Nan یا Inf- به ۰.۰ تبدیل می‌شود تا از تفسیر مقدار بازگشتی به‌عنوان یک رشته حرفی غیرعددی به‌وسیله SMSL جلوگیری شود. هنگامی که تابع SMSL این تبدیل را انجام می‌دهد، یک پیغام خطای زمان اجرا را نیز باز می‌گرداند.

1- Proper set

ج-۵ همگام‌سازی فرآیند در SMSL

SMSL همگام‌سازی فرآیند را در بین فرآیندهای SMSL یک ترتیب‌سنج فرمان واحد، از طریق عمل‌های پایه‌ای متغیر شرطی که به‌همراه قفل‌های SMSL استفاده می‌شوند، فراهم می‌کند. این عمل‌های پایه‌ای شبیه به ساختارهای ارائه‌شده برای برنامه‌نویسی چندنخی[1] در زبان برنامه‌نویسی C بر روی بسیاری از سیستم عامل‌هایی است که از نخ‌ها حمایت نمی‌کنند.

ج-۶ کانال‌های سراسری مشترک SMSL

SMSL استفاده از کانال‌های سراسری مشترک برای ارتباط بین یک فرآیند و فرآیند دیگر یا فایل را پشتیبانی می‌کند.

SMSL اجازه می‌دهد یک فرآیند SMSL کانالی را برای یک فرآیند خارجی، در حالت اشتراکی به‌طور صریح باز کند که به هر تعداد از فرآیندهای SMSL دیگر اجازه می‌دهد داده‌هایی را به آن کانال ارسال کرده، یا از آن دریافت کنند.

توانایی به اشتراک‌گذاری کانال‌ها نیازمند ارائه سازوکاری برای فنون برنامه‌نویسی همروند به‌وسیله SMSL نیز هست. SMSL این فنون را به شکل عمل‌های پایه همگام‌سازی خارجی ارائه می‌کند.

عمل‌های پایه برای ایجاد همزمانی در توابع باز کردن، خواندن، نوشتن و بستن کانال ترجیح داده می‌شوند. به‌عنوان مثال، مجبور کردن هر فرآیند SMSL به قفل کردن یک کانال مشترک به‌طور صریح، مانع از خواندن همزمان به‌وسیله یک فرآیند و نوشتن به‌وسیله دیگری می‌شود. علاوه بر این، جداکردن عمل‌های پایه همزمانی از کانال‌ها، اجازه می‌دهد که این عمل‌های پایه‌ای برای همگام‌سازی استفاده هر منبع اشتراکی مانند جدول نماد داخلی عامل یا یک فایل خارجی به‌کار روند.

اگر یک فرآیند SMSL روی کانال مسدود شده باشد، استفاده از توابع ()read، ()readln و ()write به‌وسیله سایر فرآیندها برای کانال‌های مشترک بلافاصله با شکست مواجه خواهند شد. (بدون مسدود شدن)

ج-۶-۱ تأثیر سازوکارهای کانال سراسری مشترک SMSL

توابع SMSL تضمین می‌کنند که تمام عملیات بر روی یک کانال به‌صورت متوالی انجام می‌شوند و همه فراخوانی‌های تابع SMSL تجزیه‌ناپذیر به نظر می‌رسند. برنامه‌نویس SMSL می‌تواند مطمئن باشد که خواندن‌ها و نوشتن‌های کانال فایل در فرآیندهای مختلف به‌صورت تجزیه‌ناپذیر رخ می‌دهند. قفل‌های ارائه شده در SMSL مانع از جای‌دهی[2] غیر قابل پیش‌بینی دنباله‌هایی از فراخوانی‌های خواندن و نوشتن SMSL به کانال می‌شود.

تنها مورد استثناء در متوالی‌سازی بر روی کانال‌های ایجاد شده با استفاده از تابع () popen، اجازه عملیات خواندن و نوشتن همزمان است. هنگامی‌که یک عمل نوشتن بر روی کانال در وضعیت انتظار است، عمل خواندن می‌تواند انجام شود و هنگامی که یک عمل خواندن بر روی یک کانال مسدود شده یا در حال انتظار است، عمل نوشتن

1- Multithreaded programming
2- Interleaving

می‌تواند رخ دهد. بنابراین، هر دو فرآیند SMSL خواننده و نویسنده می‌توانند بر روی یک کانال مشترک مسدود شوند.
کانال‌های فایلی که با استفاده از تابع fopen() باز شده‌اند، هرگز نمی‌تواند موجب مسدود شدن یک تابع read() یا write() در SMSL شوند. برای اِعمال متوالی‌سازی، نه فرآیند خواننده دوم و نه فرآیند نویسنده دوم نمی‌توانند مسدود شوند؛ بنابراین، تابع read()، readln() و write() دوم بر روی یک کانال فایل با شکست مواجه خواهند شد.

ج-۷ انواع داده و اشیا SMSL

SMSL دارای چهار نوع داده پایه‌ای است: عدد صحیح، اعشاری، رشته و فهرست. مقادیر هر چهار نوع پایه می‌تواند به‌عنوان رشته‌های کاراکتری دست‌کاری شود. انواع داده‌ای پیچیده می‌توانند با استفاده از ساختارهای آرایه و ساختمان ایجاد شوند.
همه انواع نحو ASN.1 به‌وسیله SMSL پشتیبانی می‌شوند.

ج-۷-۱ تبدیل بین GDMO و نوع ASN.1 و نوع اسکریپت‌گرای SMSL

جدول ج-۱ رابطه بین یک GDMO و نوع ASN.1 و نوع SMSL مشابه را نشان می‌دهد.

جدول ج-۱ - تبدیل مقدار میان ASN.1 و نوع اسکریپت گرا

گروه نوع مقدار	نوع GDMO و ASN.1	نوع اسکریپت گرا
گروه عدد صحیح	INTEGER, BOLEAN	integer
گروه عدد اعشاری	REAL	float
رشته	General string	string
گروه مجموعه	SEQUENCE, SET	object type
عبارت آرایه	SEQUENCE OF, SET OF	list

متغیرها و مقادیر به‌عنوان رشته‌ها یا اعداد، هر کجا مناسب با محتوا باشد، تفسیر می‌شوند.
اگر یک مقدار عددی (عدد صحیح یا اعشار) صفر یا رشته تهی نباشد، از دیدگاه بولی به‌عنوان درست تفسیر می‌شود. بولی‌های بازگردانده شده به‌وسیله عملگرها، ۱ برای درست و ۰ یا "" (رشته تهی) برای غلط هستند.

ج-۷-۲ ثابت‌های عددی

اگر چه نمایش داخلی یک ثابت صحیح یا اعشاری، یک رشته است، اما نیازی نیست این ثابت‌ها در اسکریپت‌های SMSL در داخل علامت نقل قول نشان داده شوند. چند مثال از ثابت‌های عدد صحیح و اعشاری SMSL عبارتند از:

x = 3; pi = 3.14159;

جدول ج-۲ – مثال‌هایی از انواع داده SMSL

نوع داده	نمایش نمونه	نمایش SMLS
integer	3	"3"

"4,5"	4.5	float
"abc"	“abc”	string
"1\n3\n5"	[1,3,5]	list
یادآوری- **“\n”** کاراکتر خط جدید است.		

ج-۷-۳ انواع داده پیچیده

یک ساختار مجموعه‌ای از انواع داده‌ای است که می‌توانند به‌منظور نمایش ساختارهای ASN.1 پیچیده با یکدیگر گروه‌بندی شوند، به‌عنوان مثال ASN.1 SETs و SEQUENCEs می‌توانند با استفاده از ساختارها ایجاد شوند. عناصر منحصربه‌فرد ساختار با عملگر "." قابل دسترسی هستند.

عبارات آرایه‌ای می‌توانند با استفاده از نوع فهرست در SMSL بیان شوند. انقیاد نام به مقدار نام شیء نگاشت می‌شود.

ج-۸ متغیرهای SMSL

متغیرهایی از هر نوع می‌توانند به‌عنوان چپ‌مقدار[1] استفاده شوند- به این معنی که می‌توان به آنها مقداری را انتساب داد. از آنجایی که همه انواع داده‌ها به‌صورت داخلی به‌عنوان رشته در نظر گرفته می‌شوند، همه آنها یک فضای نام مشترک را به اشتراک می‌گذارند. بنابراین شما نمی‌توانید برای یک متغیر عددی، یک متغیر رشته‌ای و یک متغیر فهرست از نام یکسانی استفاده کنید.

استفاده از حروف کوچک و بزرگ اهمیت دارد. ”FOO“، ”Foo“ و ”foo“ هر یک نام‌های متفاوتی هستند. اسامی باید با یک حرف یا خط زیر شروع شوند، اما می‌توانند شامل ارقام و خطوط کشیده شده زیر کلمه (”_“) نیز باشند.

بعضی شناسه‌ها، معانی از پیش تعریف‌شده دارند. کلمات کلیدی ذخیره‌شده[2] – مانند if و foreach – نمی‌توانند به‌عنوان شناسه استفاده شوند. کلمات کلیدی یا به‌طور کامل با حروف کوچک و یا به‌طور کامل با حروف بزرگ شناخته می‌شوند. علاوه بر این، ثابت‌های از پیش تعریف شده نمی‌توانند به‌عنوان شناسه استفاده شوند.

ج-۸-۱ مقداردهی پیش فرض متغیرهای SMSL

SMSL از مفهوم «اعلان‌ها[3]» برای متغیرها استفاده نمی‌کند. اولین ظهور یک شناسه منجر به افزودن آن به فهرست متغیرهای سراسری برای یک اسکریپت SMSL می‌شود. هربار که یک اسکریپت SMSL اجرا می‌شود، همه متغیرها با یک مقدار رشته تهی، مقداردهی اولیه می‌شوند. این مقدار تا زمانی که مقدار متغیر به‌وسیله برخی عملیات صریح مانند انتساب تعریف شود، تغییر نمی‌کند.

این مقداردهی اولیه پیش فرض، به رشته تهی اجازه می‌دهد که با یک متغیر به‌عنوان یک فهرست/رشته که از ابتدا تهی بوده است، یا به‌عنوان یک متغیر عددی با مقدار صفر در نظر گرفته شود. (چراکه عملگرهای محاسباتی با رشته تهی معادل صفر رفتار می‌کنند.) با این حال، اتکا به این مقدار اولیه باعث می‌شود یک پیام هشدار زمان

1 -Lvalue
2 -Reserved
3 -Declarations

اجرای SMSL در اولین استفاده از آن ایجاد شود. (اگر هشدارهای زمان اجرا فعال شده باشند.) انتساب اولیه یک مقدار "" یا صفر به ترتیب به یک متغیر فهرست/رشته یا متغیر عددی، روش بهتری محسوب می‌شود.

ج-۹ ثابت‌های از پیش تعریف‌شده SMSL

تعدادی از شناسه‌ها به‌عنوان ثابت‌ها از پیش تعریف شده‌اند، به‌طوری که می‌توانند بدون نیاز به معرفی استفاده شوند. این ثابت‌ها فقط خواندنی هستند و نباید مقدار بگیرند.

جدول ج-۳- ثابت‌های از پیش تعریف شده SMSL

ثابت	تعریف
ALARM	حالت شیء ALARM
OK	حالت شیء OK
OFFLINE	حالت شیء OFFLINE
VOID	حالت شیء VOID
EOF	ثابت وضعیت انتهای فایل
True/TRUE/True yes/YES/Yes	مقدار درست بولی (۱)
False/FALSE/False no/NO/No	مقدار نادرست بولی (۰)

ج-۱۰ الفاظ[1] رشته ای SMSL

الفاظ رشته‌ای با علامت گیومه دوتایی[2] محدود می‌شوند. الفاظ رشته‌ای می‌توانند چندخطی باشند که موجب می‌شود کاراکترهای سطر جدید بخشی از رشته شوند.

قواعد واکج‌خط[3] برای کاراکترهای فرار (مانند واکج‌خط یا نقل قول) و ایجاد کاراکترهایی مانند خط جدید یا جهش[4] استفاده می‌شوند. اینها تنها الفاظ رشته‌ای هستند که در حال حاضر در SMSL پشتیبانی شده‌اند.

جدول ج-۴- الفاظ رشته ای SMSL

ثابت	تعریف
\t	جهش

1- Literal
2 -Double quotation mark
3 -Backslash
4 -Tab

\n	خط جدید
\r	بازگشت
\b	پس‌بر (Backspace)
\A…\z	Ctrl-A … Ctrl-Z

کاراکترهای کنترلی می‌توانند در ثابت‌های رشته‌ای SMSL با استفاده از \A تا \Z برای نمایش Ctrl-A تا Ctrl-Z تعبیه شوند. همیشه یک حرف بزرگ باید استفاده شود؛ حروف کوچک به‌جز آن‌هایی که تاکنون تعریف شده‌اند (یعنی t، n و r یا b) به‌عنوان کاراکترهای فرار معتبر نیستند و یک هشدار ترجمه SMSL تولید خواهند کرد.

ج-۱۱ فهرست‌های SMSL

مقادیر فهرست با جداکردن مقادیر منحصربه‌فرد به‌وسیله ویرگول و قراردادن فهرست در قلاب‌ها[1] مشخص می‌شوند:

[1,3,5]

فهرست درون یک رشته قرار گرفته در بین علامت گیومه دوتایی که عناصرش با فضاهای خالی جدا شده‌اند، قرار می‌گیرد. فهرست به‌طور داخلی به‌صورت زیر نمایش داده می‌شود:

"1 3 5"

ج-۱۲ دستورات ساده SMSL

متداول‌ترین دستور ساده، یک عبارت است که برای تأثیرات جانبی آن ارزیابی می‌شود و به نام یک دستور عبارت خوانده می‌شود. متداول‌ترین دستور عبارت یک عمل انتساب یا یک فراخوانی تابع است. هر دستور عبارت باید با یک نقطه ویرگول[2] خاتمه یابد:

```
y = x + 10;          # assignment
set("value",50);     # function call
s = trim(s,"\t");    # both assignment and function call
```

ج-۱۳ عملگرهای SMSL

ج-۱۳-۱ عملگرهای محاسباتی

در عملگرهای محاسباتی، چنانچه اولین کاراکتر یک عملوند[3] یک رقم یا علامت منفی (-) باشد، عملوند یک عدد درنظر گرفته می‌شود. در غیر این‌صورت، عملوند یک رشته درنظر گرفته شده و به ۰ برای یک رشته تهی یا به ۱ برای یک رشته غیرتهی تبدیل می‌شود.

استفاده از یک غیر عدد در محتوای محاسباتی ممکن است منجر به تولید یک هشدار زمان اجرا شود.

---

1 -Brackets
2- Semicolon
3- Operand

جدول ج-۵- عملگرهای محاسباتی SMSL

عملگر	تعریف
+	جمع
-	تفریق
/	تقسیم
*	ضرب
%	پیمانه

ج-۱۳-۲ عملگرهای انتساب

عملگرهای انتساب برای SMSL در زیر آورده شده است.

به‌عنوان مثال، a+=b معادل a=a+b است.

جدول ج-۶- عملگرهای انتساب SMSL

عملگر	تعریف
=	انتساب
.=	الحاق به خود برای رشته‌ها
+=	جمع با خود
-=	تفریق از خود
/=	تقسیم بر خود
*=	ضرب با خود
%=	پیمانه با خود

انتساب بیتی

عملگر	تعریف
&=	و بیتی با خود
|=	یا بیتی با خود
^=	یای انحصاری با خود

انتساب شیفت

عملگر	تعریف
<<=	انتساب انتقال به چپ
>>=	انتساب انتقال به راست

عملگرهای افزایش/کاهش

به‌عنوان مثال، a++ معادل a=a+1 است.

جدول ج-۷- عملگرهای افزایش/کاهش SMSL

عملگر	تعریف
++	افزایش
--	کاهش

ج-۱۳-۳ عملگرهای بیتی

عملگرهای بیتی تعریف شده در SMSL در زیر آمده‌اند.

به‌عنوان مثال، a&=b معادل a=a&b است.

جدول ج-۸- عملگرهای بیتی SMSL

عملگر	تعریف
&	و بیتی
|	یا بیتی
&=	و بیتی با خود
|=	یا بیتی با خود
^	یا انحصاری بیتی
^=	انتساب یا انحصاری بیتی

ج-۱۳-۴ عملگرهای منطقی

عملگرهای منطقی SMSL در مورد عملوندهای خود فرض می‌کنند که درست با ۱ یا یک رشته غیرتهی، و نادرست با ۰ یا یک رشته تهی نشان داده می‌شود. با این‌حال، هنگامی‌که نتایج را بر می‌گردانند، همیشه از ۱ برای درست و از ۰ برای غلط استفاده می‌کنند.

جدول ج-۹- عملگرهای منطقی SMSL

عملگر	تعریف
&&	و منطقی
||	یا منطقی
!	نقیض منطقی

ج-۱۳-۵ عملگرهای رابطه‌ای

اگر هر دو عملوند عدد باشند، عملگرهای رابطه‌ای مقایسه‌های عددی انجام می‌دهند. در غیر این‌صورت مقایسه‌های رشته‌ای انجام می‌دهند. (یعنی مرتب سازی لغوی، واژه نامه‌ای) اگر یک رشته فقط شامل ارقام، علامت منفی یا یک

نقطه پایان جمله باشد، به‌عنوان عدد در نظر گرفته می‌شود. فضای خالی مجاز نیست. عملگرهای رابطه‌ای SMSL در نشانه‌گذاری نمایی (مانند 2.3e+27)، ثابت‌ها را به‌عنوان عدد درنظر نمی‌گیرند.

ج-۱۳-۶ عملگرهای انتقال[1]

عملگرهای انتقال، انتقال بیتی در درون بایت‌ها را انجام می‌دهند.

جدول ج-۱۰- عملگرهای رابطه‌ای SMSL

عملگر	تعریف
==	تساوی
=!	عدم تساوی
<	کمتر از
>	بیشتر از
<=	کمتر یا مساوی
>=	بزرگتر یا مساوی

جدول ج-۱۱- عملگرهای انتقال **در SMSL**

عملگر	تعریف
<<	انتقال به چپ
<<=	انتساب انتقال به چپ
>>	انتقال به راست
>>=	انتساب انتقال به راست

ج-۱۳-۷ عملگرهای رشته‌ای

SMSL دارای چندین عملگر برای دست‌کاری رشته‌ای و فهرست است.

[s1, s2, …]

عملگر فهرست، با الحاق تمامی عناصر یک فهرست جداشده به‌وسیله ویرگول، در یک رشته داخل گیومه دوتایی از اقلام جداشده با یک فضای خالی، یک فهرست می‌سازد که نمایش SMSL برای فهرست‌ها/آرایه‌ها است.

=~ (equal tilde)

عملگر =~ درالگوی =~ رشته عبارت استفاده می‌شود و مقادیر زیر را برمی‌گرداند:

- ۱ اگر الگوی عبارت منظم در رشته موجود باشد؛
- ۰ اگر الگوی عبارت منظم در رشته موجود نباشد.

اگر الگو نامعتبر باشد، SMSL یک پیغام خطای زمان اجرا را برمی‌گرداند و عملگر =~، صفر را باز می‌گرداند. (الگو موجود نیست.)

1- Shift

!~(tilde)

عملگر ~! در الگوی ~! عبارت رشته‌ای استفاده می‌شود و مقادیر زیر را باز می‌گرداند:

- ۱ اگر الگوی عبارت منظم در رشته موجود نباشد؛
- ۰ اگر الگوی عبارت منظم در رشته موجود باشد.

اگر الگو نامعتبر باشد، SMLS یک پیغام خطای زمان اجرا را باز می‌گرداند و عمل ~!، صفر را باز می‌گرداند. (الگو وجود دارد.)

ج-۱۳-۸ تقدم[1] و شرکت‌پذیری[2] عملگر SMSL

تقدم و شرکت‌پذیری عملگرهای SMSL تقریبا مشابه C و Perl است. علاوه بر عملگرهای استاندارد، عملگرهای رشته‌ای جدید – "." و [x,y,...]– به همراه تقدم‌های مربوط به آنها نیز وجود دارند.

عملگرها به ترتیب صعودی تقدم در جدول ج-۱۲ فهرست شده‌اند:

جدول ج-۱۲- تقدم و شرکت‌پذیری عملگر SMSL

تقدم عملگر	شرکت‌پذیری
= (پایین‌ترین اولویت)	راست
+=,-=,<<=,>>=,^=	راست
*=,/=,%=	راست
|=,&=	راست
||	چپ
&&	چپ
|	چپ
^	چپ
&	چپ

ادامه جدول ج-۱۲

!=,==,=~,!~	چپ
<,<=,>,>=	چپ
<< , >>	چپ
+ , - (binary)	چپ
* , / , %	چپ
. (string concat)	چپ
! , - , ++ , --	راست
()	چپ

1- Precedence
2 -Associativity

[] (بالاترین اولویت)	چپ

ج-۱۴ زبان اسکریپت‌نویسی هسته[1] SMSL

یک اسکریپت SMSL حاوی دنباله‌ای از دستورات است. فرض می‌شود تمام اشیاء ایجادشده به‌وسیله کاربر که مقداردهی اولیه نشده‌اند، تا زمانی که به‌وسیله چند عمل صریح مانند انتساب، تعریف شوند، با مقدار NULL یا صفر شروع می‌شوند.

SMLS در بیشتر قسمت‌ها یک زبان با شکل آزاد[2] است. یعنی خطوط مجبور نیستند که در یک ستون خاص، یا قبل از یک ستون خاص شروع شده یا خاتمه پیدا کنند؛ آنها می‌توانند در خط بعدی ادامه داشته باشند. از فضای خالی به‌جز برای جداسازی نشانه‌ها، صرف‌نظر می‌شود. توضیحات با کارکتر # مشخص می‌شوند و تا انتهای خط ادامه پیدا می‌کنند. به‌عنوان مثال، اینجا توضیحی درباره یک عبارت انتساب آمده است:

```
x=y;      #Assign the value of y to the variable x
```

ج-۱۴-۱ جملات ترکیبی[3] SMSL

جملات ترکیبی SMSL حاوی جملات حلقه[4] و جملات شرطی[5] هستند. در SMSL یک دنباله از جملات می‌توانند با قرارگرفتن در {} به‌عنوان یک جمله درنظر گرفته شوند. ما این جمله را یک بستک[6] جمله می‌خوانیم و آن‌را در توصیفات جمله به‌صورت {BLOCK} نشان می‌دهیم.

ج-۱۴-۱-۱ Exit

قالب

exit

توصیف

دستور exit باعث می‌شود برنامه SMSL بلافاصله خاتمه یافته و کنترل به فرآیندی که آن‌را فراخوانده است، بازگردد. هنگامی که دستور exit در یک برنامه SMSL استفاده می‌شود باید با یک نقطه ویرگول خاتمه یابد.

ج-۱۴-۱-۲ Export

قالب

export variable
export function function

1- Core scripting language
2 -Free-form
3 -Compound statements
4 -Loop statements
5 -If statements
6 -Block

پارامترها

توصیف

دستور export با استفاده از دستور requires یک متغیر یا تابع را در کتابخانه SMSL آماده صدور به کتابخانه یا برنامه SMSL دیگری می‌کند. هر دستور export می‌تواند یک متغیر یا تابع واحد را مشخص کند.

نیازی به اعلان متغیرهای سراسری و توابع قبل از دستور export نیست. دستور export نیازی به تعریف صریح یک متغیر درون کتابخانه ندارد، اما لازم است که متغیر در دستور SMSL ظاهر شود تا یک تعریف ضمنی ایجاد شود.

جایابی دستور exit

دستور export function function می‌تواند قبل یا بعد از تعریف واقعی تابع ظاهر شود. دستور export variable می‌تواند قبل یا بعد از اولین ظهور یک متغیر سراسری ظاهر شود.

دستور export می‌تواند درون یک تعریف تابع بدون هیچ اهمیت ویژه‌ای ظاهر شود.

تعریف پارامتر

variable نام یک متغیر SMSL که آماده صدور به برنامه SMSL دیگر است.

function نام یک تابع SMSL که آماده صدور به برنامه SMSL دیگر است.

خطاهای مربوط به دستور export

دستور صدور می‌تواند در موارد زیر یک خطای ترجمه ایجاد کند:

- متغیر یا تابع در کتابخاته تعریف نشده یا استفاده نشده است؛
- متغیر یا تابع، یک تابع داخلی از SMSL است؛
- متغیر، یک متغیر محلی از یک تابع تعریف‌شده کاربر در کتابخانه است؛
- متغیر یا تابع در دستور export دیگری تکرار شده است؛
- متغیر یا تابع با استفاده از دستور requires وارد شده است.

ج-۱۴-۱-۳ Foreach

قالب

```
Foreach   [list]  {BLOCK}
Foreach   unit   variable   [list]   {BLOCK}
```

توصیف

حلقه foreach بر روی متغیر فهرست و مجموعه‌ها تکرار می‌شود تا به ترتیب، برای هر عنصر فهرست BLOCK را اجرا کند.

تعریف پارامتر

List فهرستی که حاوی یک یا چند عنصر است که متغیر می‌تواند برابر آن‌ها شود.

BLOCK یک یا چند دستور که وقتی متغیر برابر یک عنصر از فهرست در نظر گرفته می‌شود، اجرا می‌شوند.

unit کنترل می‌کند که چگونه فهرست به عناصر مجزا تقسیم شود.

Valid Range

- کلمه، فرض می‌کند عناصر آرایه با فضای خالی جدا شده‌اند (فضای خالی، جهش‌ها یا\n)؛

– خط فرض می‌کند که عناصر آرایه با \n جدا شده‌اند.

پیش‌فرض اگر مشخص نشده باشد: خط

variable نام عنصری که برابر هر عنصر در فهرست قرار داده می‌شود.

مثال‌ها

مثال‌های زیر استفاده از دستور foreach را برجسته می‌کنند.

عناصر درون یک آرایه را جمع می‌زند.

```
sum = 0;
foreach elem ("1\n2\n3\n4\n5")
{
	sum += elem;
}
```

شناسه‌های ورود برای هر حساب در سامانه را فهرست می‌کند.

```
foreach user (cat ("/etc/passwd"))
{
	printf(ntharg (item, 1, ":"), "\n");
}
```

یادآوری– cat() و ntharg()، توابع داخلی در SMSL هستند.

تعداد کلمات در یک رشته را شمارش می‌کند.

```
words = 0;
foreach word w ("The cat sat on the mat.")
{
	words++;
{
```

ج-۱۴-۱-۴ Function

قالب

Function name(argument_list) {BLOCK}

توصیف

دستور function، توابع تعریف‌شده کاربر را در برنامه‌های SMSL، مشابه آنچه در زبان برنامه‌نویسی C وجود دارد، فراهم می‌کند. کلمه کلیدی function در تعریف یک تابع کاربر ضروری است. دو کلمه کلیدی دیگر، local و return اختیاری هستند:

- local، متغیرهایی را که فقط درون تابع استفاده خواهند شد، اعلان می‌کند؛
- return، خروجی تابع را که به فراخوان‌کننده بازگردانده می‌شود، شناسایی می‌کند.

توابع باید قبل از اولین استفاده از آن‌ها تعریف شوند و در یک فراخوانی تابع باید argument_list صحیح ارسال شود. فراخوانی تابع همیشه رشته کاراکتری را باز می‌گرداند که نشان‌دهنده یک رشته کاراکتری یا مقدار عددی است. (همه انواع داده‌ها در SMSL به‌عنوان رشته‌های کاراکتری نشان داده می‌شوند.)

تعریف پارامتر

name برچسب کاراکتری که برای شناسایی و فراخوانی تابع از داخل برنامه SMSL استفاده می‌شود؛ name نمی‌تواند با هیچیک از موارد زیر برابر باشد:

- تابع داخلی SMSL؛
- متغیر SMSL.

argument_list صفر یا چند متغیر SMSL که وقتی تابع برای اجرا فراخوانی می‌شود، به‌عنوان پارامتر به آن ارسال می‌شود. argument_list می‌تواند یک ورودی NULL باشد اگر هیچ متغیری به تابع ارسال نشود، یک آرگومان منفرد باشد یا چندین آرگومان جداشده با ویرگول باشد.

BLOCK یک یا چند دستور SMSL که اقدامی را که تابع انجام می‌دهد، تعریف می‌کنند.

آرگومان‌ها به‌وسیله مقدار به پارامترها ارسال می‌شوند (یعنی نسخه‌های محلی از داده‌های آرگون‌های ارسال‌شده ایجاد می‌شود) و بنابراین تغییر یک پارامتر، تأثیری بر روی مقدار آرگومان نخواهد داشت. پارامترهای یک تابع برای آن تابع محلی هستند و می‌توانند اسامی یکسان با متغیرهای سراسری داشته باشند. (یا مشابه با پارامترهای سایر توابع)

اگر تعریف یک تابع در وسط دستورات اجرایی ظاهر شود و جریان کنترل از بالا به آن تعریف برسد، از آن تعریف مانند یک توضیح، صرف‌نظر می‌شود. تنها راه ورود به بدنه یک تابع، فراخوانی صریح آن است. تعاریف تابع تنها یک تابع را تعریف می‌کنند و تا زمانی که صدا زده نشوند، فراخوانی نمی‌شوند. بنابراین می‌توان کد اجرایی را در بالا، پایین و میان تعاریف تابع قرار داد.

ج-۱۴-۱-۵ دستور return

سه روش برای خروج از یک تابع تعریف‌شده به‌وسیله کاربر وجود دارد:

- خروج با یک مقدار بازگشتی؛
- خروج بدون مقدار بازگشتی (مقدار بازگشتی = رشته NULL)؛
- پایین آمدن تا رسیدن به علامت آکولاد بسته[1] پایین (برگشت حذف شده است، مقدار بازگشتی ندارد.)

SMSL پایین آمدن تا انتهای یک تابع را به‌عنوان وضعیت خطا تفسیر نمی‌کند.

هنگامی که SMSL با دستورات return در یک تابع مواجه می‌شود که برخی از آن‌ها دارای مقادیر بازگشتی هستند، در حالی که برخی دیگر خیر، یک هشدار ترجمه، مشابه آنچه به‌وسیله مترجم C تولید می‌شود، ایجاد می‌کند. وجود چندین نقطه خروج در تابع که هر یک به روش متفاوتی خارج می‌شوند ممکن است نشان‌دهنده سردرگمی در این مسئله باشد که آیا تابع تعریف شده است تا عملی را انجام دهد یا مقداری را بازگرداند.

1- Right brace

ج-۱۴-۲ تعریف متغیرهای محلی

متغیرهای محلی تابع تعریف‌شده به‌وسیله کاربر با استفاده از کلمه کلیدی local در بدنه تابع، اعلان می‌شوند. کلمه کلیدی local یک یا چند متغیر را که در یک فهرست جداشده با ویرگول مشخص شده‌اند و با یک نقطه ویرگول خاتمه می‌یابند، اعلان می‌کند. این نام‌ها، متغیرهای محلی تابع می‌شوند. مثالی از تعاریف متغیر محلی در زیر آمده است:

```
function f()
{
        local x;
        local a,b;
        # ... Statements for the function execution.
}
```

متغیرهای محلی نمی‌توانند نامی مشابه پارامتر تابع یا متغیر محلی دیگری در همان تابع داشته باشند. اسامی متغیرهای محلی در یک تابع تأثیری بر متغیرهای یک تابع دیگر ندارند. متغیرهای محلی می‌توانند نام مشابهی با یک متغیر سراسری داشته باشند و می‌توانند به این صورت یک نام سراسری را «مخفی» کنند.

با اعلان‌های متغیر محلی به‌عنوان عبارت برخورد می‌شود و می‌توانند در هر جایی از تابع که یک عبارت معتبر است، ظاهر شوند. با این‌حال، مفهومی به نام حوزه‌های داخلی در بلاک‌های داخلی وجود ندارد و حوزه یک متغیر محلی از نقطه اعلان آن تا انتهای تابعی که درون آن قرار گرفته است (نه انتهای بلاکی که در آن قرار گرفته)، ادامه دارد.

هر زمان که وارد تابع می‌شویم، متغیرهای محلی با رشته تهی مقداردهی اولیه می‌شوند. آنها مقادیر خود از یک فراخوانی قبلی را حفظ نمی‌کنند.

بیشینه تعداد متغیرهای محلی و پارامترهای تابع در توابع تعریف‌شده به‌وسیله کاربر (به‌جز برای تابع ()main) وابسته به پیاده‌سازی است.

ج-۱۴-۳ تابع نقطه ورود

تابع نقطه ورود در SMSL تابع ()main است. اگر یک برنامه SMSL دارای تابع تعریف‌شده کاربر با نام main باشد، اجرا از اولین دستور در ()main آغاز می‌شود. برنامه SMSL هنگامی که ()main باز می‌گردد، به‌صورت عادی متوقف می‌شود. تابعی که شما به‌عنوان نقطه ورود مشخص می‌کنید مجاز به داشتن خاصیت‌های مشابه ()main است.

تابع ()main یا تابع نقطه ورود باید در برنامه SMSL سطح بالا و نه در هر کتابخانه وارد شده، تعریف شود. در هنگام تعیین موجود بودن یک تابع نقطه ورود، توابع واردشده از کتابخانه‌ها نادیده گرفته می‌شوند.

ج-۱۴-۳-۱ شروع اجرا بدون یک تابع نقطه ورود

اگر تابع ()main موجود نباشد و هیچ تابع نقطه ورودی با استفاده از گزینه e– در مترجم SMSL مشخص نشود، اجرا از اولین دستور قابل اجرایی که درون تعریف یک تابع نباشد، آغاز می‌شود. یک برنامه بدون تابع ورود به‌طور معمول تعاریف تابع را در بالا (توابع باید قبل از استفاده از آنها تعریف شوند) و دستورات قابل اجرای اصلی را بعد از آن‌ها قرار می‌دهد. مثالی معمول می‌تواند به‌صورت زیر باشد:

```
function max(x,y)
```

```
{
        if(x > y)
        {
                return x;
        }
        else
        {
                return y;
        }
}
m = max(1,2);           #Execution starts here
printf("maximum is ", m, "\n");
```

همان‌طورکه گفته شد، اجرای برنامه بلافاصله پس از تعریف تابع شروع می‌شود.

ج-۱۴-۳-۲ محدودیت‌های توابع تعریف‌شده به‌وسیله کاربر

توابع تعریف‌شده به‌وسیله کاربر، در معرض محدودیت‌های زیر هستند:

- فراخوانی‌های تابع غیربازگشتی هستند.
- توابع تعریف‌شده به‌وسیله کاربر می‌توانند سایر توابع را به‌صورت نامحدود فراخوانی کنند، به شرطی که بازگشت مستقیم یا غیرمستقیم در توالی فراخوانی‌ها وجود نداشته باشد.
- ارسال با مقدار و ارسال با ارجاع پشتیبانی شده است.
- توابع SMSL از ارسال آرگومان با ارجاع و ارسال آرگومان با مقدار پشتیبانی می‌کنند.

محدودیت‌های پارامتر و متغیر محلی

محدودیت‌های پارامتر و متغیر محلی برای توابع SMSL در این استاندارد تعریف نشده است. این محدودیت‌ها وابسته به پیاده‌سازی هستند.

تودرتویی توابع[1] مجاز نیست

SMSL تودرتویی تابع را مجاز نمی‌داند- هر تعریف تابع باید در حوزه سراسری باشد و نمی‌تواند درون تابع دیگر تعریف شود.

ج-۱۴-۴ If

قالب

```
if (expression) {BLOCK}

if (expression) {BLOCK} else {BLOCK}

if (expression) {BLOCK} elsif (expression) {BLOCK} . . . else{ BLOCK}
```

توصیف : دستور if آسان است. از آنجایی که همیشه یک دستور BLOCK به‌وسیله آکولاد محدود می‌شود، هیچگونه ابهامی درباره اینکه کدام if، elsif و else با هم هستند، وجود ندارد.

تعریف پارامتر

Expression یک عبارت SMSL که ارزیابی آن مقدار TRUE یا FALSE را باز می‌گرداند.

BLOCK یک یا چند دستور SMSL که براساس ارزیابی عبارات if یا elsif، یک‌بار اجرا می‌شوند.

مثال‌ها

مثال‌های زیر استفاده از if، elsif و else را نشان می‌دهند:

یک دستور if

```
if (x > 10)

{

        x = 10;          # don't let x get bigger than 10

}
```

یک دستور if…else

```
if (x == 0)

{

        # do something
```

1- Function nesting

```
}
else
{
        # x != 0
        # do something else
}
```

یک دستور if…elseif…else

```
if (x == 0)
{
        # do something
}
elsif (x == 1)
{
        # do something else
}
else
{
        # x !== 0 && x != 1
        # do something else
}
```

ج-۱۴-۵ Last

قالب

last

توصیف : دستور last موجب خروج اجرای SMSL از داخلی‌ترین حلقه اجرایی می‌شود. هنگامی که last در یک برنامه SMSL استفاده می‌شود باید با یک نقطه ویرگول خاتمه پیدا کند.

ج-۱۴-۶ Next

قالب

next

توصیف

دستور next بلافاصله اجرای تکرار بعدی داخلی‌ترین حلقه اجرایی را شروع می‌کند.

ج-۱۴-۷ uiresReq

قالب

requires library

توصیف

دستور requires متغیرها و توابع شناسایی‌شده در دستور export را از کتابخانه SMSL از پیش ایجادشده به درون برنامه SMSL وارد می‌کند. هر دستور requires می‌تواند یک نام کتابخانه واحد را مشخص کند.
SMSL هیچگونه دستور import صریحی ندارد؛ استفاده از دستور requires بر عمل واردکردن دلالت دارد. دستور requires فایل دودویی حاوی کتابخانه را جستجو می‌کند و تمام اطلاعات دستور export آن را می‌خواند، سپس متغیرها و/یا توابع مشخص‌شده را به درون برنامه SMSL وارد می‌کند.
در یک برنامه SMSL هر تعداد دستور requires می‌تواند ظاهر شود. تمام کتابخانه‌های مشخص‌شده در دستورات requires باید در حین ترجمه، در دسترس مترجم قرار داشته باشد.

دستور requires در کتابخانه‌های وارده

مترجم SMSL به‌طور خودکار وابستگی‌های تو در تو در کتابخانه‌های وارده را کشف می‌کند، اما هیچ‌یک از توابع و متغیرهای صادرشده دیگر که در کتابخانه موجود بوده و وابستگی تودرتو را برآورده می‌کنند را به‌طور خودکار بارگذاری نخواهد کرد. برای تضمین دسترسی به تمام متغیرها و توابع درون کتابخانه، شما باید آن کتابخانه را به‌طور صریح وارد کنید.

تعریف پارامتر

library نام کتابخانه‌ای که متغیرها و توابع مشخص‌شده در قسمت export در آن باید به درون برنامه SMSL وارد شوند.

یک دستور requires می‌تواند درون یک تعریف تابع، بدون هیچ اهمیت خاصی ظاهر شود.

دسترس‌پذیری متغیر و تابع در میان کتابخانه‌های واردشده

هنگامی‌که یک برنامه SMSL، متغیرها و توابع را از بیش از یک کتابخانه وارد می‌کند، متغیرها و توابع واردشده از یک کتابخانه می‌توانند بدون درنظر گرفتن نحوه بارگذاری کتابخانه‌ها برای ترجمه، متغیرها و توابع واردشده از سایر کتابخانه‌ها را مقداردهی کرده و استفاده کند.

خطاهای مربوط به عبارت requires

دستور requires می‌تواند در موارد زیر خطاهای ترجمه تولید کند:

- یک ارجاع به متغیر یا تابع واردشده، قبل از دستور requires که آن را وارد می‌کند، ظاهر شود. شما باید یک دستور require را قبل از اولین استفاده از متغیر یا تابع واردشده قرار دهید.
- تابع واردشده دارای نام یکسان با تابع تعریف‌شده در داخل برنامه SMSL باشد.
- نام متغیر یا تابع یکسانی از دو یا چند کتابخانه وارد شود.

ج-۱۴-۸ Switch

قالب

```
switch (variable)
{
        case a: {BLOCK}
        case b: {BLOCK}
        …
        case p,q,r: {BLOCK}
        . . .
        case n: {BLOCK}
        default: {BLOCK}
}
```

پارامترها

توصیف

دستور switch متغیر را ارزیابی کرده و بسته به مقدار عددی آن یک بلوک SMSL مشخص را اجرا می‌کند. برچسب‌های case مرتبط با مقادیر متغیر است و برای هریک از آن‌ها یک بلوک SMSL وجود دارد.

اگر مقدار متغیر خارج از محدوده مقادیر برچسب‌های case باشد، اجرا با بلوک مربوط به برچسب default ادامه پیدا می‌کند. اگر برچسب default وجود نداشته باشد، اجرا با اولین دستور پس از دستور switch ادامه پیدا می‌کند.

تعریف پارامتر

variable SMSL	نام متغیری که مقدار عددی آن، بلوک SMSLی که اجرا خواهد شد، را مشخص می‌کند.
a,b,... p,q,r,…n	مقادیر عددی که مقدار متغیری را نشان می‌دهند که منجر به اجرای بلوک مربوط می‌شود.
BLOCK	یک یا چند دستور که هرگاه مقدار case مربوط با متغیر برابر شود، اجرا می‌شوند.

توصیف

دستور switch متغیر را ارزیابی کرده و بسته به مقدار عددی آن یک بلوک SMSL مشخص را اجرا می‌کند. برچسب‌های case مرتبط با مقادیر متغیر است و برای هریک از آن‌ها یک بلوک SMSL وجود دارد.

اگر مقدار متغیر خارج از محدوده مقادیر برچسب‌های case باشد، اجرا با بلوک مربوط به برچسب default ادامه پیدا می‌کند. اگر برچسب default وجود نداشته باشد، اجرا با اولین دستور پس از دستور switch ادامه پیدا می‌کند.

دستور switch در SMSL مانند یک دنباله طولانی از دستورات if-then-else-if رفتار می‌کند. بند case یا default به‌طور مؤثر یک دستور زمان اجرا است که مقایسه با مقدار متغیر را مشخص می‌کند:

- اگر مقدار متغیر با case منطبق شود، اجرا به درون بلوک آن بند case یا default منتقل می‌شود و پس از تکمیل بلوک، اجرا با دستورات پس از کل دستور switch ادامه پیدا می‌کند. (یعنی اجرا به بند case بعدی ادامه پیدا نمی‌کند.)
- اگر مقدار متغیر با هیچ یک از caseها منطبق نشود، اجرا به بند default پرش می‌کند و اگر این بند موجود نبود، اجرا به دستور پس از دستور switch منتقل می‌شود.

هر دستوری در بلوک case از دستور switch که بخشی از یک بلوک مربوط به case یا default نیست، تنها در صورتی اجرا می‌شود که تمام برچسب‌های case بالای آن با متغیر منطبق نشوند. (یعنی، این دستور به‌عنوان بخشی از دنباله طبیعی جریان کنترل اجرا می‌شود.)

خواص[1] دستور SMSL switch در زیر آمده است:

- عبارات case می‌توانند عبارات ارزیابی‌شده به‌صورت خودکار و عبارات ثابت باشند.
- جداکننده دو نقطه که برچسب case را از بلوک اجرایی جدا می‌کند در SMSL اختیاری است.
- SMSL نیازمند است که برچسب default بعد از همه برچسب‌های case در بلوک case دستور switch قرار گیرد. اگر پس از default یک یا چند برچسب case قرار گیرد، یک خطای ترجمه باز گردانده می‌شود.
- SMSL برای برچسب‌های case تکراری در دستور switch، یک خطای ترجمه باز نمی‌گرداند. در SMSL دومین برچسب case تکراری غیرقابل دسترس است.
- SMSL اجازه می‌دهد که چندین case که یک بلوک مشترک را اجرا می‌کنند، به‌صورت فهرستی که با ویرگول جدا شده است، در یک برچسب case منفرد مشخص شوند. (برعکس، برچسب‌های انباشته‌شده در SMSL عمل نخواهند کرد.)
- اجرای یک بلوک SMSL به برچسب case و بلوک بعدی ادامه پیدا نمی‌کند. به‌محض رسیدن به آکولاد راست بسته از یک بلوک مربوط به case یا default، اجرا به انتهای دستور switch SMSL منتقل می‌شود.
- دستور SMSL switch از دستور last برای خروج از یک بلوک استفاده می‌کند. دستور last از داخلی‌ترین دستور switch یا حلقه خارج می‌شود. با این‌حال، به‌دلیل نبود «ادامه به بعدی[2]» در SMSL، نیاز اندکی به استفاده از دستور last در دستور switch وجود دارد.
- SMSL به‌محض تشخیص دو برچسب default در دستور switch واحد، یک خطای ترجمه تولید می‌کند.
- SMSL اجازه استفاده از دستورات switch تودرتو را می‌دهد.

1- Properties
2- Fall through

بلوک‌های case در زمان اجرا به ترتیب ظهور خود ارزیابی می‌شوند:

- در مورد بلوک‌ها به ترتیب case؛
- در مورد عبارات در فهرست‌های جدا شده با ویرگول از برچسب‌های case چندگانه از چپ به راست.

تمامی عبارات درون یک فهرست جداشده با ویرگول، قبل از برچسب case ارزیابی می‌شوند. حتی اگر اولین عبارت منطبق باشد نیز این ارزیابی انجام می‌شود.

این دنباله و روش ارزیابی برچسب seca می‌تواند در شرایطی که عبارتی در فهرست، متغیر دستور switch فعلی یا یک متغیر استفاده‌شده در عبارت case دیگر را تغییر دهد، یک دام خطرناک باشد. در SMSL، دستورات درون یک دستور switch که بخشی از یک بلوک نیستند (دستورات آزاد)، در صورتی که جریان اجرا به آن‌ها برسد، می‌توانند اجرا شوند و اجرا خواهند شد. شرط این‌که جریان کنترل به این دستورات برسد آن است که متغیر با هیچ یک از برچسب‌های اجرا که قبل از آن دستورات قرار دارند، منطبق نشود. SMSL هنگامی‌که یکی از دو برچسب case که در برابر متغیر ارزیابی می‌شوند، درون دیگری قرار داشته باشد، هیچ پیام خطا یا هشداری را باز نمی‌گرداند. دو مثال از این وضعیت در مثال SMSL switch زیر نشان داده شده است:

```
switch(x) {
        case 1: {
                        f1()    # Function f1 Called
                        case 2: { f2(); }       # Function f2 Unreachable
                        f3();   # Function f3 Called
                }
        default: {
                        case 4: {
                                        f4();
                                }
                }       # Function f4 called if x=4
}
```

از آنجایی که برچسب‌های case و default دستورات زمان اجرا هستند، تأثیر یک برچسب case تودرتو در دیگری این است که متغیر باید با مقدار case منطبق شده تا بلوک case اجرا شود. این یعنی متغیر باید با دو مقدار متفاوت برابر باشد! در case 1 از مثال بالا، f2 هرگز فراخوانی نخواهد شد، زیرا x نمی‌تواند با هردوی ۱ و ۲ برابر باشد.

در مورد default از مثال بالا، اگر متغیر برابر ۴ شود، f4 فراخوانی خواهد شد چراکه case 4 در دستور switch تعریف نشده است. هنگامی که متغیر برابر ۴ باشد، بلوک default اجرا می‌شود که شامل فراخوانی تابع f4 در بلوک case 4 است.

ج-۱۴-۹ While

قالب

```
while (expression) {BLOCK}
```

پارامترها

توصیف

حلقه while تازمانی که عبارت به مقدار TRUE (غیرصفر) ارزیابی می‌شود، دستورات را اجرا می‌کند.

مثال

دستورات SMSL نمونه زیر، اعداد صحیح ۱ تا ۱۰ را چاپ می‌کنند.

```
x=1;
while (x <= 10)
{
        printf(x, " ");
        x++ ;
}
printf("\n");
```

تعریف پارامتر

expression یک دستور SMSL که ارزشیابی آن TRUE یا FALSE را باز می‌گرداند.

BLOCK یک یا چند دستور SMSL که تا وقتی expression برابر TRUE ارزشیابی می‌شود، به‌طور تکراری اجرا می‌شود.

توصیف

حلقه while تا زمانی‌که expression برابر TRUE (غیرصفر) ارزیابی می‌شود، دستورات را اجرا می‌کند.

مثال

دستورات SMSL نمونه زیر، اعداد صحیح ۱ تا ۱۰ را چاپ می‌کنند.

```
x=1;
while (x <= 10)
{
        printf(x, " ");
        x++ ;
}
printf("\n");
```

ج-۱۴-۱۰ مدل شیء

SMSL مبتنی بر یک مدل ساده شی‌گرا است که امکان نگاشت از محیط مدیریت سامانه‌ها را که در GDMO (توصیه نامه CCITT Rec. X.733 | ISO/IEC 10164-4 را ببینید) توصیف شده است، فراهم می‌کند. یک شیء، ساختاری دارای خاصیت‌هایی است که خود متغیر یا شیء دیگری هستند. توابع مرتبط به یک شیء، متدهای شیء هستند.

یک کاربر SMSL می‌تواند به خاصیت‌های یک شیء با استفاده از نشانه‌گذاری زیر دسترسی پیدا کند:

objectName.propertyName

این شیء می‌تواند شیء GDMO یا شیء تعریف شده محلی دیگری باشد.

اگر این شیء، یک شیء GDMO باشد، نگاشت از خاصیت‌های GDMO به خاصیت‌های SMSL در جدول ج-۱۳ نمایش داده شده است.

جدول ج-۱۳- نگاشت میان GDMO و SMSL

خاصیت GDMO	خاصیت SMSL
برچسب کلاس	نام نوع شیء
شناسه نمونه (عدد صحیح، غیره)	مقدار متغیر نمونه شیء
مقدار اولیه در ایجاد یک نمونه	پارامترهای عملکرد "new"
نام‌های نمونه شیء مدیریت شده	نشانه‌گذاری مقدار ASN.1
نوع ASN.1 از ATTRIBUTES	نام متغیرها
برچسب ATTRIBUTE GROUPS	نام متغیر آرایه
برچسب ACTION	نام متد (تابع)
ساماندهی NOTIFICATION غیرهمزمان	سامان‌دهنده (discriminatorConstruct) onEvent

یک خاصیت می‌تواند با تخصیص مقدار به آن، به‌صورت زیر تعریف شود:

objectName.propertyName = value;

یک متد[1]، تابعی منتسب به شیء است. تابع می‌تواند به صورت زیر به شیء منتسب شود:

objectName.methodName = functionName

که در آن object یک شیء محلی موجود، method نام نسبت داده‌شده به متد و functionName نام تابع است.

متد در محتوای شیء می‌تواند به‌صورت زیر فراخوانی شود:

objectName.methodName(parameters);

ج-۱۴-۱۰-۱ پارامترهای عملکرد و اقدام GDMO

پارامترهای اقدام و عملکرد به‌صورت یک شیء نگاشت‌شده به‌وسیله قواعد توصیف‌شده در مدل شیء، به متد اقدام ارسال می‌شوند. مقادیر پارامتر بازگشتی به‌صورت یک شیء نگاشت‌شده به‌وسیله این قواعد، از متد اقدام (تابع) بازگردانده می‌شوند. قالب توصیف به‌صورت زیر است:

outputObjectName = objectName.actionName(inputObjectName);
outputObjectName = ObjectName.operationName(inputObjectName);

ج-۱۴-۱۰-۲ مرجع شیء "this"

SMSL دارای یک کلمه کلیدی ویژه به نام "this" است که می‌تواند برای ارجاع به شیء جاری استفاده شود.

this [propertyName]

ج-۱۴-۱۰-۳ ایجاد و حذف اشیاء

new

عملگری که اجازه ایجاد یک نمونه از نوع شیء تعریف‌شده به‌وسیله کاربر را می‌دهد.

ایجاد نوع شیء به دو مرحله نیاز دارد:

۱. تعریف نوع شیء با نوشتن یک تابع.

---

1 -Method

۲. ایجاد یک نمونه از شیء با new.

برای تعریف نوع شیء، یک تابع برای نوع شیء ایجاد کنید که نام، ویژگی‌ها و متدهای شیء را مشخص کند. شیء می‌تواند خاصیتی داشته باشد که خودش شیء دیگری است.

قالب

objectName = new objectType (param1[,param2] … [,paramN])

objectName نام نمونه شیء جدید است.

ObjectType تابعی است که یک نوع شیء را تعریف می‌کند.

param1 … paramN مقادیر خاصیت شیء است. این خواص، پارامترهای تعریف‌شده برای تابع objectType هستند.

یک نمونه از یک کلاس می‌تواند با عملگر "delete" حذف شود.

delete objectName

objectName نام نمونه شیء موجود است.

علاوه‌براین نوع شیء محلی می‌تواند با نوشتن یک تابع تعریف شود. سپس می‌تواند یک نمونه محلی از شیء با عملگر "new" ایجاد شود.

عبارت شیء برای نمایش انقیاد نام

اگر دو شیء دارای یک رابطه انقیاد نام باشند، عبارت زیر از آن شیء وابسته مجاز است:

superiorObjectInstance.subordinateObjectInstance

ج-۱۴-۱۰-۴ WITH

دستوری که یک شیء پیش‌فرض را برای مجموعه‌ای از دستورات تعیین می‌کند. درون این مجموعه دستورات، هر ارجاع به یک خاصیت که یک شیء را مشخص نمی‌کند، متعلق به شیء پیش‌فرض درنظر گرفته می‌شود.

نحو

```
Wwth (objectName){
        statements
}
```

ctNameobje شیء پیش‌فرض را برای استفاده به‌وسیله statements مشخص می‌کند. پرانتزهای اطراف objectName ضروری هستند. statements هر بلوک دلخواهی از دستورات است.

ج-۱۴-۱۰-۵ سامان‌دهنده رویداد **onEvent**

توصیف : یک عملگر ساماندهی رویداد برای توصیف پردازش، وقتی یک اعلان مشخص که به وسیله GDMO تعریف شده است، رخ می‌دهد.

نحو

```
onEvent(DiscriminatorConstructValue){
                statements
}
```

DiscriminatorConstructValue مقدار نوع DiscriminatorConstruct است. statement هر بلوک دلخواهی از دستورات است.

ج-۱۴-۱۰-۶ triggerParameterCount

این ماشین مجازی SMSL تعداد پارامترهای ماشه‌چکانی را به‌وسیله این متغیر پیگیری می‌کند.

ج-۱۴-۱۰-۷ triggerArgument

توصیف : triggerArgument فهرست پارامترهای ماشه‌چکانی را که یک اسکریپت SMSL می‌تواند به آن‌ها دسترسی داشته باشد، می‌پذیرد و شناسه لت راه‌اندازی را که در ادامه اسکریپت را اجرا می‌کند، باز می‌گرداند. فهرست آرگومان شامل شناسه ماشه‌چکانی و شناسه اسکریپت است، مگر این‌که ماشه‌چکانی دارای پارامتر نباشد.

نحو

triggerArgument(argument_list)

که *argument_list* می‌تواند به اندازه مقدار triggerParameterCount عنصر داشته باشد.

مثالی از اسکریپت SMSL برای مدیریت EFD

این مثال نشان می‌دهد که چگونه یک EFD می‌تواند ایجاد شده و حالت عملیاتی آن تعیین شود.

پارامترهای اسکریپت که باید ماشه‌چکانی شوند، عبارتند از شناسه اسکریپت و مقصد اعلان. اسکریپت SMSL برای دریافت صفت حالت عملیاتی EFD و برگرداندن آن به‌عنوان یک اعلان به مقصد، در زیر آمده است.

ماشه‌چکانی با پارامترهای زیر مشخص شده است:

- triggerId، شناسه ماشه‌چکانی؛
- scriptId، شناسه اسکریپتی که باید اجرا شود؛
- managerId، شناسه مقصدی که گزارش‌های رویداد باید به آن ارسال شود.

اسکریپت SMSL برای این مثال در زیر آمده است:

```
#include “CMIP.CMIP-1.h”
#include “DMI.Attribute-ASN1Module.h”

manager1 = triggerArgument(triggerId, scriptId, managerId);

/* Instance creation as CMISFilter type */
CMISFilter counterValue _GT10 = item ( greaterOrEqual ( Attribute-ASN1Module.Count,
10))
/*
* Creating instance of EFD with the following parameters.
* DiscrimnatorConstruct = ( counter > 10 ),
* administrativeState = default value, destination = manager1
*/
efd1 = new eventForwardingDiscriminator( counterValue_GT10, manager1);

triggerResult = efd1.operationalState;          /* get operational state */
printf(“operational state of event forwarding discriminator %d \n”, triggerResult);
```

مدیر مقصد به‌عنوان یک پارامتر ماشه‌چکانی عرضه شده است. ماشه‌چکانی باعث می‌شود لت راه‌اندازی نخ‌هایی را برای اجرای کلیه دستورالعمل‌های اسکریپت به ترتیب، ایجاد کند. لت راه‌اندازی پارامترهای مناسب را به نخ‌ها ارسال می‌کند. به‌عنوان مثال هنگام ایجاد edf1، لت راه‌اندازی شناسه اسکریپت و مدیر مقصد را به عنوان پارامتر به نخ ارسال می‌کند.

ج-۱۴-۱۱ BNF برای SMSL

(قراردادها: "{}" برای فراورده‌هایی[1] استفاده می‌شود که ممکن است صفر یا چند مرتبه رخ دهند؛ "[]" برای فراورده‌های اختیاری است.)

نشانه‌ها:

```
T_ELLIPSIS        : "..."
T_EQ              : "=="
T_NE              : "!="
T_REGEXP_EQ       : "=~"
T_REGEXP_NE       : "!~"
T_LEQ             : "<="
T_GEQ             : ">="
T_GT              : ">"
T_LT              : "<"
T_AND             : "&&"
T_OR              : "||"
T_NOT             : "!"
T_INC             : "++"
T_DEC             : "--"
T_PLUSEQ          : "+="
```

1 -Production

```
T_MINUSEQ          : "-="
T_MULEQ            : "*="
T_DIVEQ            : "/="
T_MODEQ            : "%="
T_BITANDEQ         : "&="
T_BITOREQ          : "|="
T_LEFT_SHIFT       : "<<"
T_RIGHT_SHIFT      : ">>"
T_RIGHT_SHIFT_ASSIGN : ">>="
T_LEFT_SHIFT_ASSIGN  : "<<="
T_XOR_ASSIGN       : "^="
```

کلمات کلیدی

```
T_IF          : "if" | "IF"
T_ELSE        : "else" | "ELSE"
T_ELSIF       : "elsif" | "ELSEIF"
T_FOREACH     : "foreach" | "FOREACH"
T_FOR         : "for" | "FOR"
T_WORD        : "word" | "WORD"
T_LINE        : "line" | "LINE"
T_NEXT        : "next" | "NEXT"
T_LAST        : "last" | "LAST"
T_WHILE        : "while" | "WHILE"
T_DO          : "do" | "DO"
T_UNTIL       : "until" | "UNTIL"
```

```
T_EXIT          : "exit" | "EXIT"

T_TRUE          : "true" | "TRUE" | "True" | "yes" | "YES" | "Yes"

T_FALSE         : "false" | "FALSE" | "False" | "no" | "NO" | "No"

T_RETURN        : "return" | "RETURN"

T_LOCAL         : "local" | "LOCAL"

T_FUNCTION      : "function" | "FUNCTION"

T_NATIVE        : "native" | "NATIVE"

T_RPC           : "rpc" | "RPC"

T_VOID          : "void"

T_REQUIRES      : "requires" | "REQUIRES"

T_EXPORT        : "export" | "EXPORT"

T_CASE          : "case" | "CASE"

T_SWITCH        : "switch" | "SWITCH"

T_DEFAULT       : "default" | "DEFAULT"
```

قواعد:

```
program : stmts

stmts : { stmt }

stmt :   expr '; '

         return  |
         if      |

         foreach |

         switch  |

         case    |

         default |

         while   |
```

```
        do_until        |
        for_loop        |
        T_LAST '; '     |
        T_NEXT '; '     |
        T_EXIT '; '     |
        function        |
        native_function         |
        rpc     |
        local_var_dec |
        export |
        requires
requires : T_REQUIRES requires_name ';'
library_name : T_STRING | T_IDENTIFIER
requires_name : library_name
export : T_EXPORT [ T_FUNCTION ] export_name ';'
export_name : T_IDENTIFIER
part : T_WORD
foreach : T_FOREACH part simple_id '('  expr ')'  '{'  stmts '}'
case_exprs : { case_expr ','  } case_expr
case_expr : expr
case : T_CASE case_exprs optional_colon '{'  stmts '}'
default : T_DEFAULT optional_colon '{' stmts '}'
optional_colon : [':' ]
switch : T_SWITCH ' (' expr ')'  '{' stmts '}'
```

```
for_loop : T_FOR '(' optional_expr ';' optional_expr ';' optional_expr ')'  '{' stmts '}'

do_until : T_DO '{' stmts '}' T_UNTIL '(' expr ')' ';'

while : T_WHILE '(' expr ')' '{' stmts '}'

void : [ T_VOID ]

native_function :
        T_NATIVE void T_FUNCTION function_name '(' func_param_list ')' ';'

rpc : T_RPC void T_FUNCTION function_name '(' func_param_list ')' ';'

function : T_FUNCTION function_name '(' func_param_list ')' '{' stmts '}'

func_param_list : param_list

function_name : T_IDENTIFIER

param_list : [ { one_param ',' } one_param ]

one_param : T_IDENTIFIER |
                T_ELLIPSIS

local_var_dec : T_LOCAL var_list ';'

var_list : { one_var ',' } one_var

one_var : T_IDENTIFIER

return : T_RETURN [ expr ] ';'

if : T_IF '(' expr ')' '{' stmts '}' opt_elsifs opt_else

opt_elsifs : { elsif }

elsif : T_ELSIF '(' expr ')' '{' stmts '}'

opt_else : [ else ]

else : T_ELSE '{' stmts '}'

optional_expr : [ expr ]

expr : unary_expr       |
        expr '+' expr                   |
        expr '-' expr                   |
        expr '*' expr                   |
        expr '/' expr                   |
```

```
        expr '%' expr                   |
        expr T_EQ expr                  |
        expr T_NE expr                  |
        expr T_REGEXP_NE expr           |
        expr T_REGEXP_EQ expr           |
        expr T_LT expr                  |
        expr T_GT expr                  |
        expr T_LEQ expr                 |
        expr T_GEQ expr                 |
        expr T_AND expr                 |
        ternary_expr                    |
        expr T_OR expr                  |
        expr '|' expr                   |
        expr '&' expr                   |
        expr '^' expr                   |
        expr T_LEFT_SHIFT expr          |
        expr T_RIGHT_SHIFT expr         |
        lvalue '=' expr                 |
        lvalue T_PLUSEQ expr            |
        lvalue T_MINUSEQ expr           |
        lvalue T_MULEQ expr             |
        lvalue T_DIVEQ expr             |
        lvalue T_BITANDEQ expr          |
        lvalue T_BITOREQ expr           |
        lvalue T_MODEQ expr             |
        lvalue T_LEFT_SHIFT_ASSIGN expr |
        lvalue T_XOR_ASSIGN expr        |
        lvalue T_RIGHT_SHIFT_ASSIGN expr |
        expr '.' expr

simple_id : T_IDENTIFIER

ternary_expr : expr '?' expr ':' expr

lvalue : simple_id

unary_expr : primary            |
        '-' unary_expr          |
        T_NOT unary_expr        |
        T_INC lvalue            |
        T_DEC lvalue

function_call_id : T_IDENTIFIER

primary : simple_id     |
        T_INT           |
        T_FLOAT         |
        T_STRING        |
        T_TRUE          |
```

```
        T_FALSE       |
         '(' expr ')'      |
        lvalue T_INC            |
        lvalue T_DEC            |
        function_call_id '(' arglist ')'

arglist : [ { expr ',' } expr ]
```

==========

Note that the definition of string should allow embedded \"within strings.

```
T_IDENTIFIER          :  [A-Za-z_][A-Za-z_0-9]*
T_STRING              :  \"[^"]*\"
T_FLOAT               :  [0-9]*"."[0-9]+
T_INT                 :  [0-9]+
```

/* Operator tokens and their precedences */

```
%right '=' T_PLUSEQ T_MINUSEQ T_MULEQ T_DIVEQ T_MODEQ T_BITANDEQ
T_BITOREQ T_LEFT_SHIFT_ASSIGN T_RIGHT_SHIFT_ASSIGN T_XOR_ASSIGN
%left '?' ':'
%left T_OR
%left T_AND
%left '|'
%left '^'
%left '&'
%left T_EQ T_NE T_REGEXP_EQ T_REGEXP_NE
%left T_LT T_GT T_LEQ T_GEQ
%left T_LEFT_SHIFT T_RIGHT_SHIFT
%left '+' '-'
%left '*' '/' '%'
%left '.'
%right T_UNARY T_NOT T_INC T_DEC
%left '('
%left '['
```

پیوست چ
(الزامی)
توابع پشتیبان SMSL

acos()

مقدار آرک‌کسینوس آرگومان را برمی‌گرداند.

قالب

acos(*cosine*)

پارامتر

پارامتر	تعریف
cosine	آرگومان کسینوس بازه معتبر: $-1 \le cosine \le 1$

توضیح

تابع acos() آرک‌کسینوس *cosine* را برمی‌گرداند که عبارت است از طول (برحسب رادیان) کمانی که کسینوس آن برابر *cosine* است.

بازه خروجی تابع acos() برابر $0 \le acos() \le p$ است. مقدار بازگشتی تابع acos() تا شش رقم اعشار دقت دارد.

asctime()

تاریخ و زمان را در قالب یک رشته از کاراکترها برمی‌گرداند.

قالب

asctime(*clock*, *format*)

پارامترها

پارامتر	تعریف
clock	یک ارجاع به ساعت یا شمارنده‌ای که مقدارش بهتر است به یک رشته از کاراکترها تبدیل شود. *clock* در اکثر موارد برابر time() است.
format	مشخصه اختیاری قالب برای رشته خروجی asctime(). مشخصه‌های فیلد زیر معتبر هستند: %a روز هفته اختصاری %A روز هفته کامل %b ماه اختصاری %B ماه کامل %c بازنمایی تاریخ و ساعت محلی %d شماره ده‌دهی روز ماه (از ۰۱ تا ۳۱) %E سال و نام Emperor/Era ترکیب‌شده %H شماره ساعت در حالت ۲۴-ساعتی در مبنای ده (از ۰۰ تا ۲۳) %I شماره ساعت در حالت ۱۲-ساعتی در مبنای ده (از ۰۱ تا ۱۲)

%j شماره روز سال در مبنای ده (از ۰۰۱ تا ۳۶۶) %m شماره ماه در مبنای ده (از ۰۱ تا ۱۲) %M شماره دقیقه در مبنای ده (از ۰۰ تا ۵۹) %n کاراکتر سطر جدید %N نام Emperor/Era %o سال Emperor/Era %p معادل AM/PM %S شماره ثانیه در مبنای ده (از ۰۰ تا ۶۱) %t کاراکتر Tab %U شماره هفته سال در مبنای ده: جمعه نخستین روز هفته است؛ همه روزهای قبل از اولین جمعه سال، در هفته صفر قرار دارند. (از ۰۰ تا ۵۳) %w شماره روز هفته در مبنای ده: جمعه نخستین روز هفته است. (از ۰۰ تا ۰۶) %W شماره هفته سال در مبنای ده: شنبه اولین روز هفته است؛ همه روزهای قبل از اولین شنبه سال، در هفته صفر قرار دارند. (از ۰۰ تا ۵۳) %x بازنمایی تاریخ محلی %X بازنمایی ساعت محلی %y شماره سال در مبنای ده بدون قرن (از ۰۰ تا ۹۹) %Y شماره سال در مبنای ده با قرن %Z نام منطقه زمانی (اگر نام منطقه زمانی وجود دارد.) % % % کاراکتر مشخصه‌های فیلد می‌تواند به شکل زیر بیان شود: [-|0] *field_specifier*.p که - فیلد را چپ چین می‌کند. (حالت پیش‌فرض راست چین است) 0 فیلد را راست چین کرده و سمت چپ آن را با ارقام صفر پر می‌کند. p. حداقل تعداد ارقام قابل نمایش برای فیلدهای در مبنای ده یا حداکثر تعداد کاراکترهای قابل نمایش برای فیلدهای الفبایی. برای فیلدهای در مبنای ده، مکان‌های کاراکتر خالی با صفرهای پیشین پر می‌شود. برای فیلدهای کاراکتری، کاراکترهای اضافی از سمت راست حذف می‌شوند. پیش‌فرض اگر مشخص نشده باشد: رشته ۲۴-کاراکتری با قالب Sun Sep 16 01:03:52 1973	

توضیح

تابع asctime() تاریخ/زمان ساعت را به‌عنوان یک رشته کاراکتری برمی‌گرداند. این تابع در زبان C معادل تابع کتابخانه‌ای asctime() است.

اگر قالب داده شده باشد، asctime() رشته تاریخ/زمان را در قالب مشخص‌شده برمی‌گرداند. مشخصه‌های فیلد استفاده‌شده در قالب، معادل مشخصه‌های استفاده‌شده در تابع کتابخانه‌ای strftime() در زبان C است .

asin()

مقدار آرک‌سینوس آرگومان را برمی‌گرداند.

قالب

asin(*sine*)

پارامتر

پارامتر	تعریف
sine	بازه معتبر: $-1 \le sine \le 1$

توضیح

تابع asin() آرک‌سینوس *sine* را برمی‌گرداند که عبارت است از طول (برحسب رادیان) کمانی که سینوس آن برابر *sine* است. بازه خروجی تابع asin() برابر $-\pi/2 \le asin() \le \pi/2$ است .

atan()

مقدار آرک‌تانژانت آرگومان را برمی‌گرداند.

قالب

atan(*tangent*)

پارامتر

پارامتر	تعریف
tangent	بازه معتبر: $-\infty \le tangent \le \infty$

توضیح

تابع atan() آرک‌تانژانت *tangent* را برمی‌گرداند که عبارت است از طول (برحسب رادیان) کمانی که تانژانت آن برابر *tangent* است.

بازه خروجی برای تابع atan() برابر $-\pi/2 \le atan() \le \pi/2$ است .

cat()

محتوای یک فایل را به‌عنوان یک تک رشته متنی برمی‌گرداند.

قالب

cat(*filename*)

پارامتر

پارامتر	تعریف
filename	نام فایلی که محتوای آن باید برگردانده شود

توضیح

تابع cat() محتوای فایل *filename* را به‌عنوان یک تک رشته، یا در صورت بروز خطا رشته NULL را برمی‌گرداند. کاراکترهای سطر جدید حفظ می‌شوند تا بتوان با استفاده از دستور foreach رشته برگردانده‌شده را به‌عنوان یک فهرست از سطرهای موجود در *filename* پردازش نمود.

مثال

دستور SMSL زیر نام کاربران فهرست‌شده در فایل کلمه رمز سامانه UNIX را فهرست می‌کنند.

```
people = cat("/etc/passwd");
foreach person (people)
{
name = ntharg(person, 1, ":");
printf("name of person is:%s", name, "\n");
}
```

ceil()

کوچکترین عدد صحیح را که از آرگومان کوچکتر نیست برمی‌گرداند.

قالب

ceil(*argument*)

پارامتر

پارامتر	تعریف
argument	آرگومان عددی که کوچکترین حد بالای عدد صحیح آن باید تعیین شود.

توضیح

تابع ()ceil کوچکترین عدد صحیح را که از *argument* کمتر نیست برمی‌گرداند که برابر کمترین حد بالای عدد صحیح برای *argument* است .

تابع ()ceil و تابع floor به‌همراه هم *argument* مانند براکت (جزء صحیح) عمل می‌کنند به‌طوری‌که موارد زیر درست هستند:

- اگر *argument* عددی صحیح است: ceil(*argument*) = *argument* = floor(*argument*).
- اگر *argument* عددی صحیح نیست: floor(*argumrnt*) < *argument* < ceil(*argument*) و ceil(*argument*) = floor(*argument*) +1.

chan_exists()

بررسی می‌کند که آیا یک فرآیند یا کانال فایل وجود دارد یا خیر.

قالب

chan_exists(*channel*)

پارامتر

پارامتر	تعریف
channel	فرآیند یا نام کانال (کانال‌های مشترک) یا تعداد کانال (کانال‌های محلی) ورودی/خروجی فایلی که قرار است بررسی شود.

توضیح

تابع ()chan_exists اگر کانال محلی یا مشترک وجود داشته باشد، مقدار ۱، و اگر وجود نداشته باشد مقدار صفر را برمی‌گرداند.

مقدار بازگشتی تابع ()chan_exists می‌تواند به‌همراه متغیرهای شرط برای همگام‌سازی یک فرآیند SMSL که تا وقتی یک فرآیند دیگر کانالی را با استفاده از تابع ()popen یا ()fopen باز کند منتظر می‌ماند، استفاده شود.

close()

یک فایل یا کانال فرآیند را می‌بندد.

قالب

close(*channel*, *flags*)

پارامترها

پارامتر	تعریف
channel	فرآیند یا نام کانال (کانال‌های مشترک) یا تعداد کانال (کانال‌های محلی) ورودی/خروجی فایلی که قرار است بسته شود.
flags	پرچم‌های بیتی اختیاری که برای کنترل اجرای عمل بسته شدن استفاده می‌شوند. بیت‌های زیر استفاده می‌شوند: Bit 1 = 1 نشان می‌دهد که هر فرآیند سامانه که با کانال مرتبط است (یعنی تابع ()fopen یا ()popen از SMSL) باید در حین بستن کانال از بین برود. Bit 2 = 1 نشان می‌دهد که کانال، حتی اگر فرآیند SMSL دیگری در انتظار یک تابع ()read، ()readln یا ()write مسدود شده باشد، باید بسته شود. Bit 2 فقط برای کانال‌های سرتاسری کاربرد دارد و به‌وسیله کانال‌های محلی نادیده گرفته می‌شود. اگر پیش‌فرض مشخص نشده باشد: بیت‌های ۱ و ۲ هر دو صفر هستند.

توضیح

تابع ()close یک کانال را به یک فرآیند یا فرمان که پیش‌تر به‌وسیله فراخوانی ()fopen یا ()popen ایجاد شده است، می‌بندد.

وقتی پرچم‌ها مشخص نشده باشند، مقدار پیش‌فرض برابر صفر است.

وقتی bit1=0، تابع ()close هیچ فرآیندی را که در نتیجه ()fopen یا ()popen ایجاد شده باشد، از بین نمی‌برد[1] و این فرآیندها مجازند به کار خود ادامه دهند. این ویژگی تابع ()close به شما اجازه می‌دهد کانالی را به یک فرآیند SMSL باز کنید، داده‌های افزوده را ارسال کنید و در حالی که به فرآیند اجازه می‌دهید کار خود را کامل کند، کانال را ببندید.

1- does not kill

وقتی bit2=1، حتی اگر فرآیند SMSL دیگری در طی یک درخواست ورودی/خروجی بر روی کانال مسدود شده باشد، تابع ()close کانال را خواهد بست. وقتی مسدود شدن اتفاق می‌افتد، ()close باعث می‌شود تابع مسدودشده فراخوانی شود و یک بازگشت خطا و شماره خطا را از فرآیندی که کانال برای آن باز شده است، دریافت کند.
اگر عمل بستن با موفقیت انجام شود، تابع ()close رشته NULL را برمی‌گرداند و در غیر این‌صورت ۱- را به‌همراه متغیر SMSL شماره خطا که مقداردهی شده است برمی‌گرداند. تابع ()close وقتی که bit2=0 و کانال، کانالی سراسری با حداقل یک فرآیند SMSL مسدود شده باشد، با شکست مواجه می‌شود.

مثال

بستن کانالی که با متغیر chan بازنمایی شده است.

```
close(chan);
```

concat()

دو رشته را به هم الحاق می‌کند.

قالب

concat(*string1*, *string2*)

توضیح

تابع concat موجب الحاق دو رشته می‌شود.
به‌عنوان مثال، concat(“ab”, “cd”) رشته “abcd” را برمی‌گرداند.

cond_signal()

به فرآیندی که در انتظار یک شرط خاص مسدود شده است، سیگنال می‌دهد.

قالب

cond_signal(*condition_variable*, *all*)

پارامترها

پارامتر	تعریف
condition_variable	نام متغیری که مسدودیت فرآیندی را که به‌وسیله تابع ()cond_wait مسدود شده است، رفع خواهد کرد.
all	مقدار غیر NULL که تابع ()cond_signal را برای رفع مسدودیت همه فرآیندهای SMSL که در انتظار برای condition_variable مسدود شده‌اند، هدایت می‌کند.

توضیح

تابع ()cond_signal می‌تواند به فرآیند SMSL دیگری که در حال حاضر به‌خاطر یک تابع ()cond_wait بر روی *condition_variable* مسدود شده است، سیگنال دهد. اگر *all* مشخص شده باشد و برابر رشته NULL نباشد، تابع ()cond_signal همه فرآیندهای SMSL را که روی condition_variable مسدود شده‌اند، فراخوانی خواهد کرد. اگر هیچ فرآیندی روی condition_variable مسدود نشده باشد، تابع ()cond_signal هیچ اثری ندارد. تابع ()cond_signal هرگز نمی‌تواند مسدود شود و همیشه رشته NULL را برمی‌گرداند.

cond_wait()

فرآیند را تا زمانی‌که یک سیگنال شرطی دریافت شود، مسدود می‌کند.

قالب

cond_wait(*condition_variable*, *lockname*, *timeout*)

پارامترها

پارامتر	تعریف
condition_variable	نام متغیری که شرط cond_wait() را خاتمه خواهد داد. *condition_variable* به‌وسیله تابع cond_signal() اعلام می‌شود.
lockname	نام قفلی که وقتی که *condition_variable* غیرمسدودکننده درست را در تابع cond_signal دریافت می‌کند، تابع cond_wait() باید تلاش کند آن را به‌دست آورد. اگر *lockname* رشته NULL باشد، تابع cond_wait() بعد از دریافت *condition_variable* تلاشی برای به‌دست آوردن قفل انجام نمی‌دهد.
timeout	تعداد ثانیه‌های انتظار برای دریافت *condition_variable* پیش از رفع مسدودیت و رهاسازی *lockname*. بازه معتبر: timeout > 0 مهلت زمانی را برحسب ثانیه مشخص می‌کند؛ timeout < 0 یک مهلت زمانی بی‌نهایت را مشخص می‌کند؛ فقط دریافت *condition_variable* می‌تواند فرآیند را رفع مسدودیت کند؛ timeout = 0 مجاز نیست و موجب بروز یک پیغام خطای زمان اجرای SMSL خواهد شد. اگر مشخص نشده باشد، مقدار پیش‌فرض برابر مهلت زمانی بی‌نهایت است.

توضیح

تابع cond_wait() فرآیند SMSL جاری را تا زمانی که *condition_variable* دریافت شود یا مهلت زمانی به انتها برسد، مسدود می‌کند. اگر زمانی که تابع cond_wait() فراخوانی می‌شود فرآیند SMSL قفل *lockname* را در اختیار داشته باشد، تابع cond_wait() قفل را آزاد می‌کند. وقتی تابع cond_wait() پارامتر *condition_variable* را دریافت می‌کند تابع cond_wait() به‌طور سریع نسبت به به‌دست آوردن قفل *lockname* اقدام می‌کند.

اگر تابع cond_wait() مقدار ۱ را برگرداند، همیشه یک قفل انحصاری روی *lockname* نگه خواهد داشت. اگر تابع cond_wait()موفق نشود، در زمان برگشت، هیچ‌گونه قفلی بر روی lockname نگه نخواهد داشت. اگر مهلت زمانی اتفاق بیفتد، تابع cond_wait() یک مقدار شکست "۰" را برمی‌گرداند، شماره خطای SMSL را با مقدار S_SMSL_TIMEOUT مقداردهی می‌کند و هیچ قفلی روی *lockname* نگه نخواهد داشت.

condition_variable

condition_variable نام متغیر شرطی است که تابع cond_wait() منتظر می‌ماند تا به‌وسیله تابع cond_signal() سیگنال‌دهی شود. نام متغیرهای شرطی دارای حوزه سراسری مشابه قفل‌ها و کانال‌های مشترک است.

هیچ‌کدام از این حوزه‌های سراسری در یکدیگر دخالتی ندارند. می‌توانید نام یکسانی را بدون هیچ‌گونه مغایرتی، برای یک قفل، یک کانال مشترک و یک متغیر شرط استفاده کنید.

lockname

در هنگام ورود به تابع ()cond_wait، فرآیند قفل *lockname* را آزاد کرده و در انتظار دریافت سیگنال، مسدود می‌شود. قفل lockname باید به‌طور معمول یک قفل انحصاری باشد که به‌وسیله این فرآیند نگه داشته شده است؛ در غیر این‌صورت، ممکن است پیغام‌های خطای زمان اجرا بروز کند. (اگرچه تابع ()cond_wait هنوز هم سعی می‌کند به پیش رفته و به‌هرحال منتظر یک سیگنال می‌شود.) همیشه تابع ()cond_wait در انتظار تابع ()cond_signal یا برای مهلت زمانی مسدود خواهد شد.

زمانی که فرآیند SMSL دیگری تابع ()cond_signal را که این فرآیند SMSL را فراخوانی کرده است، اجرا می‌کند، تابع ()cond_wait سعی خواهد کرد یک قفل انحصاری به‌دست آورد (اگر یک قفل درخواست شده باشد؛ یعنی اگر *lockname* برابر رشته تهی نباشد.) و یا بلافاصله به‌همراه قفل برمی‌گردد یا به صف انتظار برای یک قفل انحصاری بر روی *lockname* می‌پیوندد.

تأمین *lockname* شیوه رایجی است، چرا که متغیرهای شرط تقریباً همیشه به‌وسیله قفل‌ها محافظت می‌شوند. در تابع ()cond_wait، *lockname* باید به‌جای از قلم افتادن، به‌عنوان رشته NULL تأمین شود تا کدنویس SMSL را برای درنظر گرفتن این‌که آیا قفلی لازم است یا خیر، مجبور کند. متغیر الزامی *lockname* باعث کاهش تعداد خطاهایی می‌شود که به‌واسطه عدم استفاده از قفل، وقتی که قفل لازم است، روی می‌دهند.

timeout

رفتار مهلت زمانی، بدون توجه به اینکه آیا تابع ()cond_wait منتظر یک تابع ()cond_signal است یا در انتظار به‌دست آوردن *lockname* است، تغییری نمی‌کند. اگر *condition_cariable* یا *lockname* قبل از کامل شدن تابع ()cond_wait از بین رود، تابع ()cond_waitصفر را برمی‌گرداند و شماره خطای SMSL را مقداردهی می‌کند، اما هیچ قفلی روی *lockname* نگه نخواهد داشت.

متغیر *lockname* می‌تواند رشته NULL باشد که در این‌صورت این‌طور درنظر گرفته می‌شود که *condition_variable* هیچ قفل مربوطه‌ای ندارد و تابع ()cond_wait به‌محض دریافت سیگنال، بدون انتظار برای هر قفلی، بلافاصله پیغام موفقیت را برخواهد گرداند.

cos()

کسینوس آرگومان را برمی‌گرداند.

قالب

cos(*radians*)

پارامتر

پارامتر	تعریف
radians	طول (برحسب رادیان) کمانی که کسینوس آن باید تعیین شود. بازه معتبر: $-\infty \le radians \le \infty$

توضیح

تابع ()cos کسینوس radians را برمی‌گرداند.

بازه خروجی برای تابع cos() برابر $-1 \le \cos() \le 1$ است .

cosh()

کسینوس هایپربولیک آرگومان را برمی‌گرداند.

قالب

cosh(*argument*)

پارامتر

پارامتر	تعریف
argument	مقدار عددی که کسینوس هایپربولیک آن باید تعیین شود. بازه معتبر: $-\infty \le argument \le \infty$

توضیح

تابع cosh() کسینوس هایپربولیک آرگومان را برمی‌گرداند. کسینوس هایپربولیک به‌وسیله عبارت زیر تعریف می‌شود:

$$\cosh(x) = (e^{x} + e^{-x})/2$$

که e پایه لگاریم طبیعی است. (e=2.71828…) بازه خروجی برای تابع cosh() برابر $1 \le \cosh() \le \infty$ است .

date()

تاریخ و ساعت را به‌عنوان یک رشته ۲۴-کاراکتری برمی‌گرداند.

قالب

date()

توضیح

تابع date() تاریخ و ساعت جاری را به‌عنوان یک رشته ۲۴-کاراکتری در قالب زیر برمی‌گرداند:

Sun Sep 16 01:03:52 1973

تابع date() معادل تابع کتابخانه‌ای ctime(3) در زبان C است . تابع date() همچنین معادل دستور SMSL زیر است :

asctime(time());

مثال

مثال‌های زیر کاربرد تابع date() را برجسته می‌کنند.

انتساب زمان و تاریخ فعلی به یک متغیر:

today = date();

debugger()

در طول فرمان ضمیمه‌سازی از سوی اشکال‌زدای SMSL، فرآیند را به حالت تعلیق درمی‌آورد.

قالب

debugger()

توضیح

تابع ()debugger در SMSL، در طول فرمان ضمیمه‌سازی از سوی اشکال‌زدای SMSL، فرآیند SMSL جاری را به حالت تعلیق درمی‌آورد. تابع ()debugger گزینه‌های درون اشکال‌زدای SMSL را که فرآیندهای SMSL را به حالت تعلیق درمی‌آورند، کامل می‌کند. تابع ()debugger متدی سطح پایین را برای متوقف کردن یک فرآیند SMSL قبل از شروع، به‌منظور اشکال‌زدایی ارائه می‌کند.

اگرچه تغییر کد منبع SMSL برای اشکال‌زدایی یک قطعه اسکریپت خاص ممکن است راحت نباشد، تابع ()debugger روشی عمومی را ارائه می‌کند که با استفاده از آن تمام کد SMSL می‌تواند اشکال‌زدایی شود.

تنها راه برای شروع مجدد تابع SMSL که از طریق تابع debugger به حالت تعلیق درآمده است، از طریق اشکال-زدای SMSL است . اگر فرآیند SMSL در حال حاضر در حال پردازش به‌وسیله اشکال‌زدای SMSL باشد، فراخوانی تابع ()debugger مؤثر نخواهد بود. تابع ()debugger همیشه رشته NULL را برمی‌گرداند.

destroy()

یک شیء SMSL را از بین می‌برد.

قالب

destroy(*object*, *description*)

پارامترها

پارامتر	تعریف
object	شناسه الفبایی برای شیء. شناسه object زمانی که شیء ایجاد می‌شود تعیین می‌گردد.
description	رشته متنی اختیاری که می‌تواند برای توضیح اینکه چرا شیء از بین برده شده است استفاده شود. رشته متنی باید در داخل علامت گیومه قرار داده شود.

توضیح

تابع ()destroy نمونه شیء برنامه کاربردی را پاک می‌کند. تابع ()destroy، در صورت موفقیت مقدار TRUE و در صورت بروز خطا مقدار FALSE را برمی‌گرداند.

مثال

از بین بردن شیئی که نام آن در متغیر <name> قرار دارد.

```
destroy(name);
```

مقدار پیش‌فرض در صورتی که مشخص نشده باشد: رشته NULL

difference()

فهرستی از عناصر را که نسبت به یک فهرست SMSL مشخص، یکتا هستند، برمی‌گرداند.

قالب

difference(*list1, list2, list3, list4…, listn*)

پارامترها

پارامتر	تعریف

list1	فهرست SMSL که عناصر آن با عناصر همه فهرست‌های مشخص‌شده دیگر مقایسه می‌شوند.
list2…listn	یک یا تعداد بیشتری فهرست که عناصر آن‌ها با عناصر *list1* مقایسه می‌شوند.

توضیح

تابع ()difference یک فهرست SMSL با همه عناصری از *list1* را که در هیچ‌یک از فهرست‌های *list2…listn* نیستند، برمی‌گرداند. اگر *list1* فهرست NULL باشد، نتیجه نیز فهرست NULL است .

ممکن است *list1* دارای عناصر تکراری باشد. عناصر تکراری *list1*، چنانچه به‌خاطر تطبیق با بقیه فهرست‌ها در تابع ()difference حذف نشده باشند، در فهرست برگردانده‌شده به همان ترتیب و تعداد که در *list1* ظاهر شده‌اند قرار می‌گیرند.

همه عناصری که از list1 برگردانده شده‌اند در فهرست برگردانده‌شده به همان ترتیب باقی می‌مانند. اگر فهرست برگردانده‌شده برابر مجموعه NULL نباشد، عناصر مجموعه برگردانده‌شده با کاراکترهای سطر جدید از هم جدا می‌شوند؛ به‌عبارت دیگر، همه عناصر مجموعه با یک کاراکتر سطر جدید تمام می‌شوند.

execute()

فرمانی از یک نوع مشخص شده را اجرا می‌کند.

قالب

execute(*type, command, instance*)

پارامترها

پارامتر	تعریف
type	پردازشگر فرمان که باید فرمان را تفسیر و اجرا کند. بازه معتبر: انواع فرمان‌های داخلی OS یا SMSL یا یک نوع فرمان معتبر تعریف‌شده به‌وسیله کاربر
command	قوانین و نحو فرمان ارائه‌شده
instance	نمونه برنامه کاربردی که فرمان باید در برابر آن اجرا شود. مقدار پیش‌فرض در صورتی‌که مشخص نشده باشد: نمونه برنامه کاربردی که نزدیکترین نمونه اولیه فرمان است.

توضیح

تابع ()execute فرمانی از هر نوع را اجرا کرده و هر خروجی را که آن فرمان در stdout یا stderr تولید می‌کند، برمی‌گرداند. حالت فرمان در متغیر SMSL با نام exit_status ذخیره می‌شود.

مثال

```
#SQL data is returned into the buffer “data”
data = execute(“SQL”, “select * from user_objects”);
```

exists()

وجود یک شیء SMSL را درستی‌سنجی می‌کند.

قالب

exists(*object, inherit*)

پارامترها

پارامتر	تعریف
object	شناسه الفبایی شیئی که وجود آن باید تأیید شود. شناسه *object*، وقتی که شیء ایجاد می‌شود، تعیین می‌شود.
inherit	عبارت بولی که کنترل می‌کند آیا تابع exists برای تأیید وجود *object*، باید تمام سلسله مراتب وراثت را جستجو کند یا خیر. اگر *inherit*=TRUE، سلسله مراتب وراثت را جستجو نکن. اگر *inherit*=FALSE و object ارجاع به یک شیء مطلق نباشد، سلسله مراتب وراثت را جستجو کن.

توضیح

تابع ()exists، چنانچه object وجود داشته باشد TRUE و در غیراین‌صورت FALSE برمی‌گرداند. تابع ()exists در رویه‌های اکتشاف برنامه کاربردی که مشخص می‌کند آیا نمونه‌ی به‌دست آمده قبلاً کشف و در سلسله مراتب شیء نمونه‌سازی شده است یا خیر، مفید است.

مثال

```
# Check if we have created the user before
if(exists(name))
{
        printf("%f", name);
}
else
{
        printf("User name does not exist");
}
```

exp()

پایه لگاریتم طبیعی e را که به توان خاصی رسیده است، برمی‌گرداند.

قالب

exp(*exponent*)

پارامتر

پارامتر	تعریف
exponent	مقدار عددی که پایه طبیعی e باید به آن توان رسانده شود.

توضیح

تابع exp() مقدار $e^{exponent}$ را برمی‌گرداند که e پایه لگاریتم طبیعی است. (e=2.71828…)

fabs()

قدرمطلق آرگومان را برمی‌گرداند.

قالب

fabs(*argument*)

پارامتر

پارامتر	تعریف
argument	مقدار ممیز شناور که قدرمطلق آن باید تعیین شود.

توضیح

تابع fabs() قدرمطلق آرگومان را برمی‌گرداند که عبارت است از:

- *argument* اگر $argument \geq 0$؛
- *-argument* اگر $argument < 0$.

file()

اطلاعات فایل را برمی‌گرداند.

قالب

file(*filename, dummy*)

پارامترها

پارامتر	تعریف
filename	نام فایلی که تاریخ آخرین تغییرات آن باید برگردانده شود.
dummy	متغیر ساختگی که اطلاعات فایل گسترش‌یافته را به شکل زیر مشخص می‌کند: modtime atime ctime mode size numlinks type modtime تاریخ آخرین تغییر است که با تعداد ثانیه‌های سپری‌شده از نیمه‌شب ۱ ژانویه ۱۹۷۰ بیان می‌شود. atime تاریخ آخرین دسترسی است که با تعداد ثانیه‌های سپری‌شده از نیمه‌شب ۱ ژانویه ۱۹۷۰ بیان می‌شود. ctime تاریخ آخرین تغییر حالت است که با تعداد ثانیه‌های سپری‌شده از نیمه‌شب ۱ ژانویه ۱۹۷۰ بیان می‌شود. mode مجوزهای فایل است که در قالب یک عدد صحیح مبنای هشت بیان می‌شود. size طول فایل است که برحسب تعداد کاراکترها بیان می‌شود. numlinks تعداد پیوندها (لینک‌ها) به فایل مورد نظر در درون سامانه فایل است. type یک رشته کاراکتری است که نوع فایل را نشان می‌دهد. FILE فایل داده کاربر معمولی DIR فهرست راهنما SPECIAL فایل ویژه کاراکتری

	BLOCK فایل ویژه بلاکی FIFO لوله یا FIFO LINK پیوند (لینک) نمادین SOCKET سوکت (در همه platformها فراهم نیست.) UNKNOWN نوع فایل ناشناخته، که می‌تواند یک LINK یا SOCKET روی platformهایی که تابع UNIX stat() نمی‌تواند نوع را تشخیص دهد باشد؛ به‌عبارت دیگر، جایی‌که S_ISLINK یا S_ISSOCK تعریف نشده‌اند.

توضیح

تابع file() زمان آخرین تغییر فایل filename را در قالب تعداد ثانیه‌های سپری‌شده از نیمه‌شب ۱ ژانویه ۱۹۷۰ برمی‌گرداند. اگر فایل وجود ندارد، تابع file() رشته NULL را برمی‌گرداند. این تابع برای آزمودن وجود یک فایل مفید است.

یادآوری ۱– مقادیر بازگشتی تابع file() به سیستم عامل و در برخی موارد به سامانه فایل بستگی دارد. برخی platformها غیر ممکن است همه مقادیر بازگشتی را برنگردانند یا ممکن است یک یا تعداد بیشتری مقدار بازگشتی بی‌معنا داشته باشند. برای یک platform خاص، تابع file() عموماً همان اطلاعات تابع stat() زبان برنامه‌نویسی C را برمی‌گرداند.

کاربر برای خواندن فایل نیازی به مجوز ندارد، اما برای جستجو روی هر شاخه موجود در مسیر منتهی به *filename* نیاز به مجوز دارد. اگر کاربر چنین مجوزی نداشته باشد، تابع file() با شکست مواجه شده و رشته NULL را برمی‌گرداند.

مقدار *dummy* نادیده گرفته می‌شود، اما حضور آن باعث می‌شود تابع file() رشته‌ای با اطلاعات جزئی‌تر برگرداند.

مثال‌ها

مثال‌های زیر کاربرد تابع file() را برجسته می‌کند.

چاپ آخرین تاریخ تغییر فایل کلمه رمز سامانه UNIX

```
printf("%s", asctime(file("/etc/passwd")));
# modification time
```

آزمودن وجود یک فایل

```
if(file("some_file"))
{
	printf("File exists!");
}
else
{
	printf("File does not exist.");
}
```

floor()

بزرگترین عدد صحیح که از آرگومان بزرگتر نیست را برمی‌گرداند.

قالب

floor(*argument*)

پارامتر

پارامتر	تعریف
argument	آرگومان عددی که بزرگترین حد پایین عدد صحیح آن باید تعیین شود.

توضیح

تابع floor() بزرگترین عدد صحیح را که از *argument* بزرگتر نیست، برمی‌گرداند که عبارت است از بزرگترین حد پایین عدد صحیح برای *argument*.

تابع floor() و تابع ceil() با هم بر روی *argument* مانند تابع جزء صحیح (براکت) عمل می‌کنند، به‌طوری‌که عبارات زیر درست هستند:

اگر *argument* عددی صحیح است: ceil(*argument*) = *argument* = floor(*argument*)

اگر argument عددی صحیح نیست: floor(*argument*) < *argument* < ceil(*argument*) و ceil(*argument*) = floor(*argument*) + 1

fmod()

باقیمانده ممیز شناور یک عمل تقسیم را برمی‌گرداند.

قالب

fmod(*dividend*, *divisor*)

پارامترها

پارامتر	تعریف
dividend	مقدار ممیز شناوری که باقیمانده آن بعد از تقسیم شدن بر *divisor* برگردانده خواهد شد.
divisor	مقدار ممیز شناوری که *dividend* را تقسیم خواهد کرد.

توضیح

تابع fmod() باقیمانده ممیز شناور تقسیم (*dividend*)/(*divisor*) را برمی‌گرداند.

fopen()

یک کانال SMSL به فایل باز می‌کند.

قالب

fopen(*filename*, *mode*)

پارامترها

پارامتر	تعریف
filename	نام فایلی که کانال SMSL باید به آن باز شود
mode	مد دسترسی به فایل. بازه‌های معتبر: r بازکردن برای خواندن w کاهش به طول صفر برای نوشتن یا ایجاد فایل برای نوشتن a بازکردن برای افزودن به انتهای فایل یا ایجاد برای نوشتن

	rb بازکردن فایل در مبنای دو برای خواندن wb کاهش فایل در مبنای دو به طول صفر برای نوشتن یا ایجاد فایل در مبنای دو برای نوشتن ab بازکردن فایل در مبنای دو برای افزودن به انتهای فایل یا ایجاد فایل در مبنای دو برای نوشتن r+ بازکردن برای خواندن و نوشتن (به‌روزرسانی) w+ کاهش به طول صفر برای خواندن و نوشتن یا ایجاد برای خواندن و نوشتن a+ بازکردن برای خواندن و نوشتن در انتهای فایل یا ایجاد فایل برای خواندن و نوشتن r+b بازکردن فایل در مبنای دو برای خواندن و نوشتن (به‌روزرسانی) w+b کاهش فایل در مبنای دو به طول صفر برای خواندن و نوشتن یا ایجاد فایل در مبنای دو برای خواندن و نوشتن a+b بازکردن فایل در مبنای دو برای خواندن و نوشتن در انتهای فایل یا ایجاد فایل در مبنای دو برای خواندن و نوشتن

توضیح

تابع ()fopenکانالی را به filename باز می‌کند که دسترسی به آن را از داخل یک فرآیند SMSL فراهم می‌سازد. توابع ()read، ()write، ()get_chan_info، ()share و ()close در کانال‌هایی که به فایل‌ها باز شده‌اند، قابل استفاده‌اند.

تابع ()fopen، هنگامی که به‌وسیله سیستم عامل زیرین پشتیبانی شود، بررسی‌های امنیتی را انجام می‌دهد تا برای تشخیص دهد که آیا نام کاربر فرآیند فراخواننده، دارای مجوز برای آن درخواست است یا خیر.

اگر تابع ()fopen موفق باشد، شماره کانال SMSL مربوط به filename را برمی‌گرداند. شکست در بازکردن filename، برای مثال یک مشکل سیستم عامل یا حالت غیرمعتبر، شماره خطای SMSL را مقداردهی کرده و موجب می‌شود تابع fopen بدون تلاش برای بازکردن فایل، رشته NULL را برگرداند.

پشتیبانی از دسترسی به فایل در مبنای دو

تابع ()fopen اجازه استفاده از مدهای در مبنای دو را با استفاده از کاراکتر a b فراهم می‌کند، اگرچه در داخل یک فرآیند SMSL راهی برای نوشتن هیچ شکلی از داده در مبنای دو به‌جز رشته‌های کاراکتری وجود ندارد.

هم‌ترازی[1] یک فایل بعد از هر عمل فایل SMSL

توابع SMSL اطمینان حاصل می‌کنند که یک فایل بعد از هر عملیات، هم‌تراز شود تا اشکال معروف انجام یک نوشتن-سپس-خواندن[2] یا خواندن-سپس-نوشتن[3] بدون یک fseek یا fflush مداخله‌کننده، در عملیات فایل SMSL رخ ندهد.

fseek()

نشانگر موقعیت فایل را تنظیم می‌کند.

1- Flush
2- Write-then-read
3- Read-then-write

قالب

fseek(*channel, offset, whence*)

پارامترها

پارامتر	تعریف
channel	کانال ورودی/خروجی فایل که وقتی فایل به‌وسیله تابع fopen باز شده برگردانده شده است.
offset	تعداد بایت‌هایی که باید به whence افزوده شوند تا موقعیت فایل به‌دست آید.
whence	نقطه استاندارد درون فایل که offset به آن افزوده می‌شود تا موقعیت جدید فایل به‌دست آید. بازه معتبر: یکی از مقادیر عدد صحیح زیر: ۰ SEEK_SET، ابتدای فایل ۱ SEEK_CURR، موقعیت جاری فایل ۲ SEEK_END، انتهای فایل

توضیح

تابع ()fseek نشانگر موقعیت filename را به موقعیت بایتی که شماره آن برابر whence به‌علاوه offset است، تنظیم می‌کند. اگر whence نامعتبر باشد، تابع ()fseek مقدار پیش‌فرض whence=0 را استفاده و یک خطای زمان اجرا صادر کرده، اما عمل جستجوی فایل را کامل می‌کند.

یادآوری ۲- فراخوانی تابع ()fseek روی فایل‌های در مبنای دو با whence=2 (SEEK_END) بر روی همه platformها به شکل معناداری پشتیبانی نمی‌شود.

تابع ()fseek در صورت موفقیت مقدار ۰ و در صورت شکست مقدار ۱- را برمی‌گرداند. برای یک کانال نامعتبر، یعنی برای یک کانال لوله به‌جای کانال فایل، تابع ()fseek مقدار ۱- را برگردانده و یک خطای زمان اجرا را صادر و متغیر شماره خطای SMSL را مقداردهی می‌کند.

حالت fseek و افزودن به انتهای فایل

استفاده از تابع ()fseek برای تغییر نشانگر موقعیت فایل در فایلی که درحالت افزودن باز شده است، یعنی حالت‌های a، ab یا a+، از نوشتن در انتهای فایل با استفاده از تابع ()write جلوگیری نخواهند کرد.

مثال

زبان SMSL معادلی برای تابع ()rewind در زبان C ندارد، اما استفاده از تابع ()fseek در مثال زیر معادل تابع rewind(*channel*) در زبان C است :

```
fseek(channel, 0, 0);
```

ftell()

نشانگر موقعیت فایل را برمی‌گرداند.

قالب

ftell(channel)

پارامتر

پارامتر	تعریف
channel	کانال ورودی/خروجی فایل که وقتی فایل به‌وسیله تابع fopen باز شده برگردانده شده است

توضیح

تابع ()ftell نشانگر موقعیت فایل را در قالب تعداد بایت‌ها (به شکل یک عدد صحیح) از ابتدای فایل برمی‌گرداند. برای یک کانال نامعتبر، یعنی یک کانال لوله به‌جای کانال فایل، تابع ()ftell مقدار ۱- را برگردانده، خطای زمان اجرا را صادر کرده و متغیر شماره خطای SMSL را مقداردهی می‌کند.

نتیجه مرسوم هر دو نسخه C و SMSL تابع ftell تعداد کاراکترهای نوشته شده در فایل یا خوانده شده از فایل است ، به‌جز در مورد platformهایی که تبدیل CR/LF U به سطر جدید را روی فایل‌های متنی انجام می‌دهند. با اینحال، مقدار تابع ()ftell بعد از اجرای یک تابع ()fseek به انتهای فایل، به‌طور معمول برابر تعداد کل کاراکترها در فایل است .

توابع SMSL زیر، نشانگر موقعیت فایل را تغییر می‌دهند:

- fopen();
- fseek();
- read();
- readln();
- write();

تابع ()get_chan_info نشانگر موقعیت فایل را تغییر نمی‌دهد. تابع ()close باعث نامعتبر شدن *channel* می‌شود.

full_discovery()

اینکه آیا فرآیند در حال حاضر در یک چرخه اکتشاف کامل است را تایید می‌کند.

قالب

full_discovery()

توضیح

تابع ()full_discovery، چنانچه کد SMSL دربرگیرنده آن یک کد کشف برنامه کاربردی باشد و در حال حاضر در چرخه اکتشاف کامل باشد، مقدار TRUE را برمی‌گرداند. در غیراین‌صورت، تابع ()full_discovery مقدار FALSE را برمی‌گرداند.

چرخه اکتشاف کامل پس از آن که حافظه نهان فرآیند عامل تجدید شد، انجام می‌شود. بنابراین این پرچم نشان می‌دهد که آیا حافظه نهان فرآیند از آخرین زمانی که کد اجرا شده، تجدید شده است یا خیر.

مثال

مثال زیر این مطلب را که آیا کد SMSL در یک چرخه اکتشاف کامل است، آزمایش می‌کند و چنانچه پاسخ منفی باشد کد را خاتمه می‌دهد.

```
# If we are not in a full discovery cycle
# we van exit immediately
if(!full_discovery())
{
```

```
        exit();
}
```

get()

مقدار فعلی یک متغیر را برمی‌گرداند.

قالب

get(*variable*)

پارامتر

پارامتر	تعریف
variable	نام متغیری که مقدار فعلی آن برگردانده خواهد شد

توضیح

تابع ()get مقدار فعلی متغیر را برمی‌گرداند. اگر *variable* یک نام وابسته بوده و در متن اسکریپت SMSL وجود نداشته باشد، تابع ()get به‌طور متوالی متن هر یک از نمونه‌های اولیه را جستجو می‌کند تا *variable* پیدا شده یا جستجو در متن کامپیوتر با شکست مواجه شود.

مثال

مثال زیر وضعیت فعلی پایگاه داده از نوع RDB با نام Dev را برمی‌گرداند.

```
get("/RDB/Dev/status");
```

get_chan_info()

اطلاعات وضعیت را از یک فایل SMSL یا کانال فرآیند برمی‌گرداند.

قالب

get_chan_info(*channel, flags*)

پارامترها

پارامتر	تعریف
channel	نام کانال (کانال‌های مشترک) یا تعداد کانال(کانال‌های محلی) برای کانالی که وضعیت آن باید گزارش شود، یا "" برای نشان دادن اینکه تمام کانال‌ها باید گزارش شوند. (تحت کنترل پرچم‌ها)
flags	مقدار عدد صحیح نمایش‌دهنده دو پرچم در مبنای دو که خروج اطلاعات کانال سراسری و محلی را به شکل زیر کنترل می‌کند: ۱ فقط کانال‌های سراسری ۲ فقط کانال‌های محلی ۳ هم کانال‌های سراسری و هم کانال‌های محلی مقدار پیش‌فرض در صورتی‌که مشخص نشده باشد: ۱

توضیح

تابع ()get_chan_info اطلاعات کانال را به شکل یک رشته در قالب زیر برمی‌گرداند:

name status details type scope read_pid read_name write_pid write_name

اعلام دستورِ get_chan_info(“”); باعث می‌شود تابع get_chan_info() توصیف همه کانال‌های سراسری مشترک را برگرداند. این توصیف‌ها به شکل یک فهرست که عناصر آن با سطر جدید از هم جدا شده‌اند، قالب‌دهی می‌شوند. هر کانال در یک سطر نوشته می‌شود.

تعریف فیلد

name یکی از موارد زیر را شامل می‌شود:

- نام کانال - scope=SHARED.
- شماره کانال محلی - scope=LOCAL .
- همه کانال‌های مشترک و محلی - scope=”” .

وضعیت OPEN یا CLOSED است.

details یکی از موارد زیر را شامل می‌شود:

- کانال fopen() – نام فایلی که باز شده است، یا NONE اگر هیچ فایلی باز نشده است؛
- کانال popen() – شناسه فرآیند فرآیند سیستم عامل خارجی که کانال به آن متصل شده است؛ یا
- ۱- اگر فرآیند خاتمه یافته است.

نوع PIPE یا FILE است.

دامنه شمول SHARED یا LOCAL است.

read_pid یکی از موارد زیر را شامل می‌شود:

- شناسه فرآیند از فرآیند SMSL که منتظر خواندن از کانال می‌باشد.
- ۱- اگر هیچ فرآیندی منتظر نیست.

read_name یکی از موارد زیر را شامل می‌شود:

- نام فرآیندی که منتظر خواندن از کانال است.
- NONE اگر هیچ فرآیندی منتظر نیست.
- UNAVAILABLE اگر فرآیندی وجود دارد اما نام آن فراهم نیست.

write_pid یکی از موارد زیر را شامل می‌شود:

- شناسه فرآیند از فرآیند SMSL که منتظر نوشتن در کانال است.
- اگر هیچ فرآیندی منتظر نیست.

write_name یکی از موارد زیر را شامل می‌شود:

- نام فرآیندی که منتظر نوشتن در کانال است.
- NONE اگر هیچ فرآیندی منتظر نیست.
- UNAVAILABLE اگر فرآیندی وجود دارد اما نام آن در دسترس نیست.

اعلام دستور get_chan_info(“”, flags); باعث می‌شود تابع get_chan_info() همه کانال‌های محلی و سراسری فرآیند SMSL جاری را تحت کنترل مقدار پرچم‌ها برگرداند. به یاد داشته باشید که توابع get_chan_info() که در زیر ذکر شده است معادل هم هستند و برای هر دو فهرست کانال‌های سراسری برگردانده می‌شود:

get_chan_info(“”);

get_chan_info(“”, 1);

تابع ()get_chan_info، اگر پرچم‌ها غیرعددی باشند یا بزرگتر از صفر نباشند یک اخطار زمان اجرا را تولید می‌کند، اما هیچ مقداری به متغیر شماره خطای SMSL داده نمی‌شود. اگر کانال برابر رشته تهی نباشد، مفسر SMSL پرچم‌ها را بدون تولید خطا نادیده می‌گیرد.

تابع ()get_chan_info برای هر کانالی همه فیلدها را برمی‌گرداند، حتی اگر آنها برای آن کانال خاص قابل اعمال نباشند.

تابع ()get_chan_info در موارد زیر رشته NULL را برمی‌گرداند:

- هیچ کانال سراسری و/یا کانال محلی برای مقادیر داده شده پرچم‌ها وجود نداشته باشد؛
- یک شماره یا نام بد کانال را دریافت کند.

در این‌صورت، تابع ()get_chan_info همچنین متغیر شماره خطای SMSL را مقداردهی می‌کند.

getenv()

مقدار رشته‌ای یک متغیر محیطی SMSL را برمی‌گرداند.

قالب

getenv(*variable*)

پارامتر

پارامتر	تعریف
variable	نام شیئی که مقدار آن باید برگردانده شود.

توضیح

تابع ()getenv مقدار رشته‌ای یک متغیر در محیط کد SMSL را برمی‌گرداند. مقدار متغیر می‌تواند از هریک از مکان‌های زیر برگردانده شود:

- متغیرهای محیطی تعریف‌شده پارامتر؛
- متغیرهای محیطی تعریف‌شده برنامه کاربردی؛
- محیط تعریف‌شده کامپیوتر؛
- محیط عامل در آغاز اجرای آن.

تابع ()getenv جدول‌های محیطی را به ترتیب بیان‌شده جستجو می‌کند و مقدار اولین متغیر منطبق را برمی‌گرداند. اگر متغیر مورد نظر تعریف نشده باشد تابع ()getenv رشته NULL را برمی‌گرداند و متغیر شماره خطای SMSL را به یک مقدار غیرصفر مقداردهی می‌کند. اگر تابع ()getenv موفق باشد، مقدار متغیر را برگردانده و به متغیر شماره خطای SMSL مقدار صفر می‌دهد. بنابراین، می‌توانید از متغیر شماره خطا برای متمایز کردن یک متغیر تعریف‌نشده از متغیری که با رشته NULL مقداردهی شده استفاده کنید.

مثال

این مثال SMSL تابعی ارائه می‌کند که وجود یک متغیر محیطی را می‌آزماید.

```
function is_environment_var_defined(name)
{
```

```
        getenv(name); # Throw away return value of getenv
        return (errno == 0); # errno is only zero if name is defined
}
```

get_vars()

فهرست متغیرهای یک شیء SMSL را برمی‌گرداند.

قالب

get_vars(*object, showchildren*)

پارامترها

پارامتر	تعریف
object	نام اختیاری شیئی که متغیرهای آن باید فهرست شوند. مقدار پیش‌فرض در صورتی‌که مشخص نشده باشد: شیء فعلی
showchildren	پرچم اختیاری که مقدار غیرصفر آن نشان می‌دهد get_vars() باید زیراشیای شیء را هم فهرست کند. مقدار پیش‌فرض اگر مشخص نشده باشد: صفر است.

توضیح

اگر شیئی مشخص نشده باشد، تابع get_vars() فهرستی از متغیرهای شیء مورد نظر، یا شیء جاری را برمی-گرداند. اگر شیء مورد نظر وجود نداشته باشد تابع get_vars() رشته NULL را برمی‌گرداند.

فهرست متغیرهای شیء برحسب حروف الفبا به‌صورت صعودی مرتب شده است.

grep()

سطرهایی از یک بلوک متنی که با یک عبارت باقاعده منطبق است را برمی‌گرداند.

قالب

grep(*regular_expression, text, v*)

پارامترها

پارامتر	تعریف
regular_expression	دنباله کاراکتری که الگویی را تعریف می‌کند که تابع grep باید در متن آن را جستجو کند. پارامتر *regular_expression* با عبارات باقاعده‌ای که در فرمان ed(1) از Unix و توضیح regexp(5) از Unix تعریف شده‌اند، مطابقت دارد. در زیر، خلاصه‌ای کوتاه از چند کاراکتر عبارت باقاعده ذکر شده است: ^ ابتدای سطر >\ ابتدای یک کلمه $ انتهای سطر <\ انتهای یک کلمه . با هر تک کاراکتری تطبیق داده می‌شود. * با صفر یا تعدادی تکرار آنچه پس از آن می‌آید تطبیق داده می‌شود. [] با هریک از کاراکترهای داخل آن تطبیق داده می‌شود. [^] با هر کاراکتری به‌جز کاراکترهای داخل آن تطبیق داده می‌شود.
text	متنی که باید برای یافتن تطبیق‌های regular_expression مورد جستجو قرار گیرد. پارامتر text می‌تواند یک رشته متنی که در علامت نقل قول قرار گرفته باشد، یا یک یا چند فرمان SMSL که

	متنی را به‌عنوان خروجی تولید می‌کنند، باشد.
v	کاراکتر v خروجی تابع grep() را معکوس می‌کند و باعث می‌شود همه سطرهایی از متن که شامل تطبیقی برای regular_expression نیستند، در خروجی قرار گیرند. این پرچم شبیه پرچم grep v– در Unix است.

توضیح

تابع grep() فهرستی از سطرهای متن که با *regular_expression* مطابقت دارند را برمی‌گرداند.

اگر یک کاراکتر به‌جز v به‌عنوان پرچم معکوس تطبیق تابع grep() ارسال شود، مفسر SMSL پیغام خطای زمان اجرا را برمی‌گرداند.

مثال

```
# search for “martin” substring in /etc/passwd
all_lines = cat(“/etc/passwd”);
# fill a buffer with passwd
matching_line = grep(“martin”, all_lines);
```

history()

اطلاعات تاریخچه را از پایگاه داده تاریخچه برمی‌گرداند.

قالب

history(*parameter*, *format*, *number*)

پارامترها

پارامتر	تعریف
parameter	نام شیئی که تاریخچه آن باید برگردانده شود. عبارت ““ نشان‌دهنده پارامتر فعلی است. پارامتر می‌تواند یکی از موارد زیر باشد: مسیر مطلق مثل ”/APP/INST/PARAM“، مسیر وابسته مثل ”.“ یا ”../DIFFERENTINST/PARAM“. برای تاریخچه پارامتر فعلی، پارامتر می‌تواند ““ یا ”.“ باشد. مقدار پیش‌فرض اگر مشخص نشده باشد: پارامتر فعلی است.
format	رشته کاراکتری اختیاری درون نماد نقل قول که قالب هر ورودی تابع history() را مشخص می‌کند. مقادیر معتبر: n تعداد نقاط داده در دسترس را به‌عنوان اولین مقدار در فهرست بازگشتی برمی‌گرداند. t شامل یک مهر زمانی برای هر ورودی در فهرست بازگشتی است . v شامل مقدار هر ورودی تاریخچه در فهرست بازگشتی است . مقدار پیش‌فرض اگر مشخص نشده باشد: ntv است.
number	مقدار عددی اختیاری که تعداد ورودی‌هایی را که تابع history برخواهد گرداند، محدود می‌کند. مقدار پیش‌فرض اگر مشخص نشده باشد: ۵۰ است.

توضیح

تابع ()history به پایگاه داده‌ی تاریخچه پارامتر دسترسی پیدا کرده و فهرستی شامل تعداد نقاط داده موجود را به‌همراه تعدادی ورودی برمی‌گرداند.
اگر یک کاراکتر با قالب بد فراهم شود، تابع ()history رشته تهی را برگردانده، یک خطای زمان اجرا را ایجاد و متغیر شماره خطای SMSL را مقداردهی می‌کند.
به‌دلیل پیش‌فرض‌های فراهم‌شده در تابع ()history، مشخصه‌های تابع زیر معادل هستند:

history(parameter)
history(parameter, “ntv”, 50)

قالب خروجی تاریخچه

تابع ()history، بسته به این‌که کدام‌یک از پرچم‌های قالب تنظیم شده‌اند، یکی از قالب‌های زیر را برخواهد گرداند:

- *number_entries*\n اگر پرچم n تنظیم شده باشد؛
- value\n، time\n، یا value time\n اگر پرچم‌های v، t، و vt تنظیم شده باشند.

تابع ()history مقادیر یک ورودی را با فضای خالی و ورودی‌های بعدی را با کاراکتر سطر جدید جدا می‌کند.
می‌توانید از تابع (nthline(list1 برای به‌دست آوردن تعداد نقاط از سرصفحه فهرست و همچنین برای استخراج ورودی‌ها استفاده کنید. در صورت لزوم، ورودی‌ها می‌توانند با استفاده از تابع ()ntharg به مقادیر ساعت و داده تقسیم شوند. اگر یکی از دو پرچم t یا v غایب باشد، ورودی‌ها مقادیر تنها خواهند بود.

index()

موقعیت شروع یک رشته درون رشته دیگر را برمی‌گرداند.

قالب

index(*text, string*)

پارامترها

پارامتر	تعریف
text	متنی که باید برای وقوع رشته، مورد جستجو قرار گیرد. *text* می‌تواند یک رشته متنی درون نماد نقل قول، یا یک یا چند فرمان SMSL باشد که متنی را به‌عنوان خروجی تولید می‌کنند.
string	یک یا چند کاراکتر درون نماد نقل قول که باید در داخل رشته مکان‌یابی شوند.

توضیح

تابع ()index موقعیت داخل متن که رشته‌ی مورد نظر از آنجا شروع می‌شود را برمی‌گرداند، یا اگر رشته در متن وجود نداشته باشد مقدار صفر را برمی‌گرداند. اولین موقعیت در متن، موقعیت ۱ است.

int()

بزرگترین عدد صحیح که از آرگومان بزرگتر نیست را برمی‌گرداند.

قالب

int(*number*)

پارامتر

پارامتر	تعریف
number	مقدار عددی یا متغیر عددی

توضیح

تابع ()int بزرگترین عدد صحیح که از *number* بزرگتر نیست را برمی‌گرداند.

internal()

یک فرمان داخلی عامل را پردازش می‌کند.

قالب

internal(*command*)

پارامتر

پارامتر	تعریف
command	رشته متنی یعنی فرمانی است که عامل باید پردازش کند.

توضیح

تابع ()internal موجب می‌شود عامل، فرمان رشته‌ای را به‌طور داخلی و به یک نوع خاص-platform پردازش کند. تابع ()internal، در صورت موفقیت، خروجی فرمان را برمی‌گرداند. در صورت عدم موفقیت یا در صورتی که فرمان بر روی platform خاص پشتیبانی نشده باشد، تابع ()internall رشته NULL را برگردانده و متغیر شماره خطای SMSL را با E_SMSL_NOT_SUPPORTEd مقداردهی می‌کند.

تابع ()internal با هدف استفاده در امر نظارت بر کاربر و فرآیند و درخواست‌های منابع که می‌توانند در داخل عامل مدیریت شوند، طراحی شده است. در این موارد، تابع ()internalبسیار کاراتر از فراخوانی یک فرمان اختصاصی است که نیازمند فراخوانی مفسر SMSL یا مفسر فرمان دیگری است.

intersection()

فهرستی شامل عناصری که بین همه فهرست‌های مشخص‌شده مشترک هستند را برمی‌گرداند.

قالب

intersection(*list1, list2, list3, list4…listn*)

پارامترها

پارامتر	تعریف
list1…listn	دو یا تعداد بیشتری فهرست SMSL که برای عناصر مشترک مورد ارزیابی قرار می‌گیرند.

توضیح

تابع ()intersection فهرستی شامل عناصری که در همه فهرست‌های list1…listn ظاهر می‌شوند را برمی‌گرداند.

فهرست بازگردانده‌شده خوش-تعریف نیست و اگر عناصر تکراری در همه فهرست‌ها به تعداد و ترتیب یکسانی وجود داشته باشند، عناصر تکراری خواهد داشت. عناصر فهرست بازگردانده‌شده به‌وسیله تابع ()intersection به همان ترتیبی که در list1 بوده‌اند، ظاهر می‌شوند.
اگر فهرستی برابر فهرست NULL باشد، مقدار بازگشتی فهرست NULL است؛ در غیر این‌صورت همه ورودی‌های درون فهرست بازگردانده‌شده، به کاراکتر سطر جدیدختم می‌شوند.

isnumber()

اینکه آیا یک رشته، بازنمایی عددی معتبری است یا خیر را تأیید می‌کند.

قالب

isnumber(*string*)

پارامتر

پارامتر	تعریف
string	رشته‌ای که باید نسبت به رعایت معیار یک بازنمایی عددی ارزیابی شود.

توضیح

اگر متغیر مورد نظر رشته‌ای باشد که به‌عنوان یک عدد معتبر به‌حساب آید، تابع ()isnumber یک مقدار بولی ۱، و در غیراینصورت صفر را برمی‌گرداند.
یک عدد معتبر برای هر کاراکتر موجود در متغیر، فقط دارای رقم، نقطه یا علامت منها است . فضای خالی یا هر کاراکتر نامعتبر دیگری در هر جای رشته، موجب می‌شود تابع ()isnumber مقدار صفر را برگرداند. تابع ()isnumber برای رشته NULL مقدار صفر را برمی‌گرداند.

is_var()

وجود یک متغیر شیء SMSL را تأیید می‌کند.

قالب

is_var(*object*)

پارامتر

پارامتر	تعریف
object	نام شیئی که باید به‌عنوان یک متغیر تأیید شود.

توضیح

اگر شیء موجود بوده و یک متغیر باشد، تابع ()is_var مقدار TRUE را برمی‌گرداند. در موارد زیر، تابع ()is_var مقدار FALSE را برمی‌گرداند:

- شیء وجود نداشته باشد؛
- شیء وجود داشته باشد اما یک متغیر نباشد. (یعنی یک نمونه برنامه کاربردی یا یک پارامتر باشد.)

length()

تعداد کاراکترهای یک رشته را برمی‌گرداند.

قالب

length(*text*)

پارامتر

پارامتر	تعریف
text	متنی که تعداد سطرهای آن (یعنی تعداد کاراکترهای سطرجدید آن) باید شمارش شود. پارامتر *text* می‌تواند رشته متنی درون نماد نقل قول، یا یک یا تعداد بیشتری فرمان SMSL باشد که به‌عنوان خروجی، متنی تولید می‌کنند.

توضیح

تابع ()length طول متن را بر حسب تعداد کاراکترها، با در نظر گرفتن سطر جدیدها برمی‌گرداند.

lines()

تعداد سطرهای یک رشته را برمی‌گرداند.

قالب

lines(*text*)

پارامتر

پارامتر	تعریف
text	متنی که تعداد سطرهای آن (یعنی تعداد کاراکترهای سطر جدید آن) باید شمارش شود. متن می‌تواند نام یک فایل متنی، یک رشته متنی درون نماد نقل قول، یا یک یا تعداد بیشتری فرمان SMSL باشد که به‌عنوان خروجی، متنی تولید می‌کنند.

توضیح

تابع ()lines تعداد کاراکترهای سطر جدید متن را برمی‌گرداند. تابع ()lines برای بازگرداندن طول یک فهرست مفید است، چرا که اقلام موجود در فهرست یا کاراکترهای سطر جدید از هم جدا می‌شوند.

lock()

یک قفل فرآیند SMSL را بدست می‌آورد.

قالب

lock(*lockname, mode, timeout*)

پارامترها

پارامتر	تعریف
lockname	نام قفلی که باید به‌دست آید.
mode	کنترل اختیاری مجاز تحت قفل مورد نظر بازه معتبر:

	s اشتراکی r خواننده w نویسنده x انحصاری پیش‌فرض در صورتی که مشخص نشده باشد: x است. اگر نخستین حرف مد یکی از حروف s، r، w یا x نباشد، خطای زمان اجرا رخ داده و مقدار مد به پیش‌فرض x (انحصاری) تغییر می‌کند.
timeout	مقدار عدد صحیح اختیاری که تعداد ثانیه‌ها را قبل از آنکه درخواست قفل منقضی شود، مشخص می‌کند. بازه معتبر: timeout > 0 برابر تعداد ثانیه‌هایی است که طی آن درخواست قفل معتبر است. timeout = 0 به‌معنای درخواست قفل غیر-مسدودکننده است. timeout < 0 به‌معنای مهلت زمانی بی‌نهایت است. (یعنی انتظار تا زمانی که قفل رها شود.) مقدار پیش‌فرض اگر مشخص نشده باشد: مهلت زمانی بی‌نهایت است.

توضیح

تابع ()lock قفلی با نام *lockname* را درخواست می‌کند. مد درخواست، یکی از دو حالت دسترسی اشتراکی (*reader*) یا انحصاری (*writer*) تحت قفل را مشخص می‌کند. مهلت زمانی اختیاری مشخص‌کننده‌ی تعداد ثانیه-هایی است که درخواست معتبر است.

رفتار پیش‌فرض تابع ()lock آن است که یک قفل انحصاری با یک مهلت زمانی بی‌نهایت درخواست می‌کند. تابع ()lock در صورت موفقیت مقدار ۱ و در صورت شکست مقدار صفر را برمی‌گرداند.

قفل‌ها و SMSL

نام قفل‌ها نسبت به عامل، سراسری هستند؛ بنابراین:

- همه فرآیندهای SMSL جدول یکسانی از قفل‌ها را به اشتراک می‌گذارند؛
- فرآیندهای SMSL متفاوت می‌توانند نام قفل پارامترها را به اشتراک بگذارند تا اعمال همزمان را انجام دهند.

راهی برای تحمیل حوزه نامگذاری قفل وجود ندارد. توصیه می‌شود نام قفل‌ها در برنامه‌های SMSL به‌طور یکتا با استفاده از نام برنامه کاربردی کدگذاری شود. این روش از تصادم‌های احتمالی با دیگر برنامه‌های SMSL جلوگیری خواهد کرد.

درخواست‌های قفل اشتراکی

آن دسته از درخواست‌های قفل اشتراکی که برای قفلی که در حال حاضر در مد اشتراکی باشند، تأیید می‌شوند – مگر آنکه یک درخواست نوشتن در حال انتظار وجود داشته باشد. اولویت دادن به یک درخواست نوشتن در حال انتظار، از ایجاد قحطی‌زدگی[1] در سازوکار قفل برای فرآیندهای نوشتن پیشگیری می‌کند.

1- Starving

درخواست برای قفل‌هایی که در حال حاضر نگه‌داشته شده‌اند

امکان درخواست قفلی که در حال حاضر آن را نگه‌داشته‌اید، وجود دارد اگرچه شیوه خوبی نیست:

- درخواست قفلی که در حال حاضر آن را نگه‌داشته‌اید نادیده گرفته می‌شود؛
- درخواست یک قفل اشتراکی روی قفلی که در حال حاضر آن را با دسترسی انحصاری نگه‌داشته‌اید نیز نادیده گرفته می‌شود.

درخواست ارتقاء به دسترسی انحصاری برای قفلی که در حال حاضر به‌عنوان اشتراکی نگه‌داشته شده است، به‌شرطی که شما تنها فرآیند در حال استفاده از قفل اشتراکی باشید، با موفقیت انجام می‌شود و قفل ارتقا می‌یابد. اگر شما تنها فرآیند در حال استفاده از قفل نباشید، تابع ()lock بلافاصله مقدار صفر را در مد غیرمسدودکننده برمی‌گرداند. (بدون توجه به مقدار مهلت زمانی، زیرا مسدود کردن موجب ایجاد بن‌بست فوری به‌دلیل انتظار برای خودتان می‌شود!)

به‌جای استفاده از این ویژگی ارتقاء، توصیه می‌شود تابع ()unlock را فراخوانی کنید تا پیش از تلاش برای اکتساب قفل انحصاری جدید، قفل اشتراکی را رها کنید. ردگیری قفل با استفاده از متغیر SMSLDebug قابل انجام است. متغیر SMSLDebug می‌تواند در اشکال‌زدایی تعامل‌های مربوط به قفل چندپردازشی مفید باشد.

شکست تابع قفل

تابع ()lock در موارد زیر می‌تواند با شکست مواجه شود:

- اگر یک درخواست غیرمسدودکننده با شکست مواجه شود؛
- اگر پیش از آنکه قفل فراهم شود، مهلت زمانی بگذرد.

تابع ()lock در موارد زیر می‌تواند برای یک مهلت زمانی بی‌نهایت با شکست مواجه شود:

- اگر یک درخواست ارتقای مورد خاص پاسخ داده شود؛
- سامانه، عملی برای اصلاح بن‌بست خارجی اجرا کرده باشد.

loge()

لگاریتم طبیعی آرگومان را برمی‌گرداند.

قالب

loge(*argument*)

پارامتر

پارامتر	تعریف
argument	مقدار عددی که لگاریتم طبیعی آن باید تعیین شود. بازه معتبر: argument > 0

توضیح

تابع ()loge لگاریتم طبیعی آرگومان را با توجه به پایه لگاریتم طبیعی e = 2.71828… برمی‌گرداند. بازه خروجی برای تابع ()loge برابر -∞ < ()loge < ∞ است.

log10()

لگاریتم آرگومان را در پایه ۱۰ برمی‌گرداند.

قالب

log10(*argument*)

پارامتر

پارامتر	تعریف
argument	مقدار عددی که لگاریتم آن در پایه ۱۰ باید تعیین شود. بازه معتبر: argument > 0

توضیح

تابع log10() لگاریتم آرگومان را با توجه به پایه ۱۰ برمی‌گرداند.

بازه خروجی تابع log10() برابر $-\infty < \log10() < \infty$ است .

ntharg()

فهرستی قالب‌دهی شده را که فیلدهایی از یک رشته متنی را شامل می‌شود را برمی‌گرداند.

قالب

ntharg(*text*, *arguments*, *delimiters*, separator)

پارامترها

پارامتر	تعریف
text	متنی که باید به‌وسیله تابع ntharg() به فیلدهایی جداسازی شود. پارامتر *text* می‌تواند یک رشته متنی درون نماد نقل قول، یا یک یا تعداد بیشتری فرمان SMSL باشد که به‌عنوان خروجی، متنی تولید می‌کنند .
arguments	فهرستی از اعداد صحیح که شماره فیلدهایی که ntharg() باید در هر سطر از متن به‌دنبال آن‌ها بگردد را مشخص می‌کند. فیلدها به شکل زیر مشخص می‌شوند: *x,y* فیلد x و فیلد y ؛ *x-y* همه فیلدها از x تا y (با y)؛ x– همه فیلدها از ۱ تا x (با x)؛ -x همه فیلدها از x تا کاراکتر سطر جدید (با سطر جدیدسطر جدید)
delimiters	یک یا تعداد بیشتری کاراکتر که ntharg() باید در زمان بررسی متن، با آن‌ها به‌عنوان جداکننده فیلدها برخورد کند. مقدار پیش‌فرض در صورتی که مشخص نشده باشد: فضای خالی و \t (کاراکتر *tab*)
separator	کاراکتر اختیاری که باید بین هر فیلد از خروجی ntharg() قرار داده شود. مقدار پیش‌فرض در صورتی که مشخص نشده باشد: کاراکتر سطر جدید ()

توضیح

تابع ntharg() آرگومان‌های موجود در متن را برمی‌گرداند.

تابع ntharg() به‌طور طبیعی، هر سطر از متن را به‌عنوان یک فهرست از فیلدها که با فضای سفید (spcae یا tab) از هم جدا شده‌اند، تفسیر می‌کند. اگر پارامتر delimiters داده شده باشد، فهرست کاراکترهایی را مشخص می‌کند که ntharg() باید با آنها به‌عنوان جداکننده‌های فیلدها برخورد کند. تابع ntharg() به‌طور طبیعی فیلدهای انتخاب‌شده

را در قالب یک فهرست که عناصر آن با سطر جدید از هم جدا شده‌اند، برمی‌گرداند. اگر پارامتر separator داده شده باشد، جداکننده‌ای را مشخص می‌کند که باید بین اقلام موجود در فهرست بازگردانده‌شده قرار گیرد.

یادآوری ۳- تفاوت بین تابع ()ntharg و تابع ()nthargf به‌شرح زیر است:

- تابع ()ntharg با هر جداکننده‌ای که به‌دنبال یک کاراکتر غیرجداکننده بیاید، به‌عنوان انتهای یک فیلد برخورد می‌کند. تابع ()ntharg دو یا تعداد بیشتری جداکننده مجاور هم را به‌عنوان یک جداکننده تنها تفسیر می‌کند.
- تابع ()nthargf با هر جداکننده‌ای به‌عنوان انتهای یک فیلد برخورد می‌کند. تابع ()nthargf دو یا تعداد بیشتری جداکننده مجاور هم را به‌عنوان جداکننده یک یا تعداد بیشتری رشته NULL که محتوای آن‌ها می‌تواند مورد درخواست قرارگرفته و برگردانده شود، تفسیر می‌کند.

مثال

مثال زیر، نام ورود و فهرست راهنمای home مربوط به هر کاربر فهرست‌شده در فایل کلمه رمز سامانه UNIX را چاپ می‌کند.

```
foreach user (cat("/etc/passwd"))
{
        printf(ntharg(user, "1,6", ":", "\t"), "\n");
}
```

nthargf()

رشته‌ای قالب‌دهی‌شده که فیلدهایی از یک رشته متنی را شامل می‌شود را برمی‌گرداند.

قالب

nthargf(*text*, *arguments*, *delimiters*, separator)

پارامترها

پارامتر	تعریف
text	متنی که باید به‌وسیله تابع ()nthargf به فیلدهایی جداسازی شود. پارامتر *text* می‌تواند یک رشته متنی درون نماد نقل قول، یا یک یا تعداد بیشتری فرمان SMSL باشد که به‌عنوان خروجی، متنی تولید می‌کنند .
arguments	فهرستی از اعداد صحیح که شماره فیلدهایی را که ()nthargf باید در هر سطر از متن به‌دنبال آن‌ها بگردد، مشخص می‌کند. فیلدها به شکل زیر مشخص می‌شوند: *x,y* فیلد x و فیلد y ؛ *x-y* همه فیلدها از x تا y (با y)؛ x– همه فیلدها از ۱ تا x (با x)؛ x- همه فیلدها از x تا کاراکتر سطر جدید (با سطر جدیدسطر جدید)
delimiters	یک یا تعداد بیشتری کاراکتر که ()nthargf باید در زمان بررسی متن، با آن‌ها به‌عنوان جداکننده فیلدها برخورد کند. تابع ()nthargf هر وقوع جداکننده‌ها را به‌عنوان جداکننده یک فیلد در نظر می‌گیرد. تابع ()nthargf دو یا تعداد بیشتری جداکننده مجاور را به‌عنوان جدا کردن یک یا تعداد بیشتری فیلد NULL تفسیر می‌کند. مقدار پیش‌فرض در صورتی که مشخص نشده باشد: فضای خالی و \t (کاراکتر *tab*)
separator	کاراکتر اختیاری که باید بین هر فیلد از خروجی ()nthargf قرار داده شود. مقدار پیش‌فرض در صورتی که مشخص نشده باشد: کاراکتر سطر جدید () است.

توضیح

تابع ()nthargf آرگومان‌های موجود در متن را برمی‌گرداند.

تابع ()nthargf به‌طور طبیعی، هر سطر از متن را به‌عنوان یک فهرست از فیلدها که با فضای سفید (spcae یا tab) از هم جدا شده‌اند، تفسیر می‌کند. اگر پارامتر delimiters داده شده باشد، فهرست کاراکترهایی را مشخص می‌کند که ()nthargf باید با آنها به‌عنوان جداکننده‌های فیلدها برخورد کند. تابع ()nthargf به‌طور طبیعی فیلدهای انتخاب‌شده را در قالب یک فهرست که عناصر آن با سطر جدید از هم جدا شده‌اند، برمی‌گرداند. اگر پارامتر separator داده شده باشد، جداکننده‌ای را مشخص می‌کند که باید بین اقلام موجود در فهرست بازگردانده‌شده قرار گیرد.

یادآوری ۴- تفاوت بین تابع ()nthargf و تابع ()ntharg به‌شرح زیر است:

- تابع ()nthargf با هر جداکننده‌ای به‌عنوان انتهای یک فیلد برخورد می‌کند. تابع ()nthargf دو یا تعداد بیشتری جداکننده مجاور هم را به‌عنوان جداکننده یک یا تعداد بیشتری رشته NULL که محتوای آن‌ها می‌تواند مورد درخواست قرار گرفته و برگردانده شود، تفسیر می‌کند.
- تابع ()ntharg با هر جداکننده‌ای که به‌دنبال یک کاراکتر غیرجداکننده بیاید به‌عنوان انتهای یک فیلد برخورد می‌کند. تابع ()ntharg دو یا تعداد بیشتری جداکننده مجاور هم را به‌عنوان یک جداکننده تنها تفسیر می‌کند.

nthline()

سطرهای مشخص شده‌ای از یک رشته متنی را برمی‌گرداند.

قالب

nthline(*text, lines, separator*)

پارامترها

پارامتر	تعریف
text	متنی که باید به‌وسیله تابع ()nthline به تعدادی سطر جداسازی شود. پارامتر *text* می‌تواند یک رشته متنی درون نماد نقل قول، یا یک یا تعداد بیشتری فرمان SMSL باشد که به‌عنوان خروجی، متنی تولید می‌کنند.
lines	فهرستی از اعداد صحیح را که شماره سطرهایی که ()nthline باید در متن به‌دنبال آن‌ها بگردد، مشخص می‌کند. سطرها به شکل زیر مشخص می‌شوند: *x,y* سطر x و سطر y ؛ *x-y* همه سطرها از x تا y (با y)؛ x– همه سطرها از ۱ تا x (با x)؛ -x همه سطرها از x تا کاراکتر EOF (با EOF)
separator	کاراکتر اختیاری که باید بین هر فیلد از خروجی ()nthline قرار داده شود. مقدار پیش‌فرض در صورتی که مشخص نشده باشد: کاراکتر سطر جدید () است.

توضیح

تابع ()nthline سطرهای متن را به شکلی که با کاراکترهای سطر جدید از هم جدا شده‌اند، برمی‌گرداند. اگر یک جداکننده مشخص کنید، تابع ()nthline از آن جداکننده برای جداکردن سطرها استفاده خواهد کرد.

یادآوری ۵- تفاوت بین تابع ()nthlinef و تابع ()nthline به‌شرح زیر است:

- تابع ()nthlinef با هر کاراکتر سطر جدید به‌عنوان یک سطر برخورد می‌کند.
- تابع ()nthline فقط با یک سطر غیرتهی (یعنی سطری یا یک کاراکتر غیر سطر جدید که کاراکتر سطر جدید به‌دنبال آن می‌آید) به‌عنوان یک سطر برخورد می‌کند.

مثال

کد SMSL زیر، پنج فرآیند بالاتر در حال اجرا روی یک سامانه UNIX را چاپ می‌کند.

```
# print the top five processes
printf("%s", nthline(system("ps -eaf"), "2-6"));
```

nthlinef()

سطرهای مشخص شده‌ای از یک رشته متنی را برمی‌گرداند.

قالب

nthlinef(*text, lines, separator*)

پارامترها

پارامتر	تعریف
text	متنی که باید به‌وسیله تابع ()nthlinef به تعدادی سطر جداسازی شود. پارامتر *text* می‌تواند یک رشته متنی درون نماد نقل قول، یا یک یا تعداد بیشتری فرمان SMSL باشد که به‌عنوان خروجی، متنی تولید می‌کنند.
lines	فهرستی از اعداد صحیح را که شماره سطرهایی که ()nthlinef باید در متن به‌دنبال آن‌ها بگردد، مشخص می‌کند. سطرها به شکل زیر مشخص می‌شوند: *x,y* سطر x و سطر y ؛ *x-y* همه سطرها از x تا y (با y)؛ x– همه سطرها از ۱ تا x (با x)؛ -x همه سطرها از x تا کاراکتر EOF (با EOF)
separator	کاراکتر اختیاری که باید بین هر فیلد از خروجی ()nthlinef قرار داده شود. مقدار پیش‌فرض در صورتی که مشخص نشده باشد: کاراکتر سطر جدید () است.

توضیح

تابع ()nthlinef سطرهای متن را به شکلی که با کاراکترهای سطر جدید از هم جدا شده‌اند، برمی‌گرداند. اگر یک جداکننده مشخص کنید، تابع ()nthlinef از آن جداکننده برای جداکردن سطرها استفاده خواهد کرد.

یادآوری ۶- تفاوت بین تابع ()nthlinef و تابع ()nthline به‌شرح زیر است:

- تابع ()nthlinef با هر کاراکتر سطر جدید به‌عنوان یک سطر برخورد می‌کند.
- تابع ()nthline فقط با یک سطر غیرتهی (یعنی سطری یا یک کاراکتر غیر سطر جدیدکه کاراکتر سطر جدید به‌دنبال آن می‌آید) به‌عنوان یک سطر برخورد می‌کند.

توصیه می‌شود برای سازگاری با دیگر توابع SMSL، از تابع ()nthlinef استفاده کنید.

مثال

کد SMSL زیر، پنج فرآیند بالا را در حال اجرا روی یک سامانه UNIX را چاپ می‌کند.

```
# print the top five processes
printf("%s", nthlinef(system("ps -eaf"), "2-6"));
```

popen()

یک کانال SMSL به یک فرآیند باز می‌کند.

قالب

popen(*type, command, instance*)

پارامترها

پارامتر	تعریف
type	پردازشگر فرمان که باید فرمان را تفسیر و اجرا کند. بازه معتبر: انواع فرمان‌های توکار OS یا SMSL یا یک نوع فرمان معتبر تعریف‌شده به‌وسیله کاربر
command	قوانین و نحو دستور ارائه‌شده
instance	نمونه برنامه کاربردی که فرمان باید در برابر آن اجرا شود. مقدار پیش‌فرض در صورتی که مشخص نشده باشد: نمونه برنامه کاربردی که نزدیکترین نمونه‌ی اولیه دستور است.

توضیح

تابع ()popenفرآیندی برای اجرای یک فرمان از نوع تعریف‌شده ایجاد می‌کند و یک شماره کانال که می‌تواند پس از آن برای خواندن خروجی فرمان یا نوشتن پیغام‌هایی به فرمان مورد استفاده قرار گیرد را برمی‌گرداند.

تابع ()popen در صورت بروز خطا مقدار ۱- را برمی‌گرداند.

pow()

یک عدد را به توان می‌رساند.

قالب

pow(*base, exponent*)

پارامترها

پارامتر	تعریف
base	مقدار عددی که باید به تعداد *exponent* مرتبه در خودش ضرب شود.
exponent	مقدار عددی که تعداد دفعاتی که base باید در خودش ضرب شود را نشان می‌دهد. اگر $base \geq 0$ آنگاه *exponent* باید مثبت باشد و اگر $base < 0$ آنگاه *exponent* باید عددی صحیح باشد.

توضیح

تابع ()pow حاصل توان‌رسانی پایه به توان exponent، یا توان پایه را برمی‌گرداند.

بازه خروجی برای تابع ()pow برابر $-\infty < pow() < \infty$ است.

printf()

متنی را که طبق مشخصات رویه تابع کتابخانه‌ای ()printf در زبان C قالب‌دهی شده است، چاپ می‌کند.

قالب

printf(*format*)

پارامتر

پارامتر	تعریف
format	متن، نام متغیرها و کاراکترهای کنترلی که محتوا و قالب خروجی به کامپیوتر یا پنجره خروجی کار را مشخص می‌کند. پارامتر format اجازه استفاده از کاراکترهای تبدیل زیر را فراهم می‌کند: %d عدد در مبنای ده علامت‌دار (مشابه %i) %i عدد در مبنای ده علامت‌دار (مشابه %d) %u عدد در مبنای ده بدون علامت %o عدد در مبنای هشت بدون علامت %x عدد در مبنای شانزده بدون علامت با استفاده از abcdef %X عدد در مبنای شانزده بدون علامت با استفاده از ABCDEF %c کاراکتر بدون علامت %s رشته کاراکتری %e عدد اعشاری دقت مضاعف به شکل drddde±ddd که هر d یک رقم و r کاراکتر مبنا است. %E عدد اعشاری دقت مضاعف به شکل drdddE±ddd که هر d یک رقم و r کاراکتر مبنا است. %f عدد در مبنای ده به شکل dddrddd که هر d یک رقم و r کاراکتر مبنا است. %g اگر توان بعد از تبدیل کمتر از ۴- است، به شیوه %e و در غیراین‌صورت به شیوه %f چاپ می‌کند. %G به شیوه %E و با درنظر گرفتن اینکه دقت، تعداد ارقام با ارزش را مشخص می‌کند چاپ می‌کند. %N ارقام را در گروه‌های سه‌تایی گروه‌بندی کرده و آنها را با ویرگول‌هایی که در سمت راست رشته آغاز می‌شوند، جدا می‌کند. %% یک کاراکتر % را چاپ می‌کند. پارامتر format از کاراکترهای تبدیل اشاره‌گرهای C، یعنی %p و %n پشتیبانی نمی‌کند. پارامتر format اجازه استفاده از پرچم‌های زیر را می‌دهد: - چپ چین کردن و پرکردن سمت راست با فضاهای خالی + نمایش علامت به‌علاوه وقتی مقدار بزرگتر از صفر است. ۰ پرکردن با استفاده از صفر، اگر هیچ نحوه پرکردن دیگری مشخص نشده باشد. # معنای یک تبدیل را تغییر می‌دهد: 0x یا 0X را به تبدیل‌های %x و %X اضافه می‌کند. همیشه کاراکتر مبنا را به تبدیل‌های %e، %E، %f، %g و %G اضافه می‌کند. صفرهای انتهایی را در تبدیل‌های %g و %G حفظ می‌کند. پرچم # بر تبدیل‌های %c، %d، %s یا %i اثری ندارد.

توضیح

تابع ()printf خروجی را روی رایانه یا پنجره خروجی کار با استفاده از قالب‌دهی مشابه تابع استاندارد printf زبان C نمایش می‌دهد.
قالب بد یا قالبی که برای زبان C معتبر است اما برای تابع ()printf معتبر نیست، منجر به یک خطای زمان اجرای SMSL می‌شود که متغیر شماره خطای SMSL را مقداردهی می‌کند.
مقدار بازگشتی تابع ()printf همیشه رشته پوچ است.

قراردادهای C که به‌وسیله تابع printf زبان SMSL پشتیبانی نمی‌شوند

تابع ()printf از قراردادهای استفاده از ستاره (*) به‌عنوان نشانگر طول فیلد یا دقت در C پشتیبانی نمی‌کند. تابع ()printf از کاراکترهای تبدیل p% و n% پشتیبانی نمی‌کند.
اصلاح‌کننده‌های طول h، l و L معتبر نیستند و به‌وسیله تابع printf نادیده گرفته می‌شوند.
تبدیل قالب‌های تابع ()printf به‌طور مستقیم به رویه تابع کتابخانه‌ای ()printf در زبان C روی هر platform هدایت می‌شود. خروجی برای ویژگی‌های قالب‌دهی مبهم ممکن است روی platformها متفاوت باشد.

تفاوت‌های تبدیل بین رویه printf زبان C و تابع printf زبان SMSL

تبدیل‌های قالب، بین C استاندارد و SMSL، معنای یکسانی دارند، اما مفهوم انوع متغیر بین آن دو تفاوت دارد.
زبان SMSL فقط انواع رشته‌ای را برای متغیرهایش پشتیبانی می‌کند و بنابراین آرگومان‌های رشته‌ای تابع ()printf طبق روش مناسبی برای تبدیل قالب، تبدیل می‌شوند:

- قالب‌های عدد صحیح مانند d% رشته را به اعداد صحیح علامت‌دار تبدیل می‌کنند.
- قالب‌های عددی غیر عدد صحیح مانند f% به مقادیر ممیز شناور تبدیل می‌کنند.
- c% معادل ASCII آرگومان عددی‌اش، یا اولین کاراکتر آرگومانش را برای آرگومان‌های غیرعددی چاپ می‌کند. (اعمال c% به "65" موجب چاپ 'A' و اعمال c% به "AB" موجب چاپ 'A' خواهد شد.)
- s% منجر به هیچ تبدیلی نمی‌شود.
- %% نیازی به آرگومان ندارد.

تبدیل قالب N%

تابع ()printf یک توسعه C غیراستاندارد را فراهم می‌کند- تبدیل N%. تبدیل N% یک رشته عددی را به‌طوری پیش‌پردازش کرده که هر گروه سه رقمی، با شروع از سمت راست رشته، به‌وسیله یک ویرگول از گروه دیگر جدا می‌شود.
به‌عنوان مثال، تبدیل N% منجر به تبدیل‌های زیر می‌شود:

1234 ➔ 1,234 12345 ➔ 12,345 123456 ➔ 123,456

تبدیل‌های N%، در ضمن جستجو به‌دنبال اولین دنباله ارقام، علامت منهای ابتدای عدد و فضاهای خالی را نادیده می‌گیرد، بنابراین N% می‌تواند به مقادیر منفی اعمال شود. اگر هیچ رقمی بعد از کاراکترهای ردشده پیدا نشود، آرگومان چاپ‌شده تغییری نمی‌کند.
تبدیل N% فقط اولین دنباله ارقام را تغییر می‌دهد. به‌عنوان مثال، تبدیل N% اعداد ممیز شناور نظیر 1234.1234 را تغییر داده و بدون تغییر دادن دنباله ارقام سمت راست نقطه ممیز، به 1,234.1234 تبدیل می‌کند.

تابع ()printf، به‌عنوان قسمتی از تبدیل N%، با استفاده از مشخصه‌های طول فیلد و دقت که در قالب فراهم شده است، یک تبدیل s% انجام می‌دهد. تابع ()printf رشته حاصل از ترکیب تبدیل‌های N% و s% را چاپ می‌کند. به‌دلیل تبدیل تعبیه‌شده s%، طول فیلد و دقت، تحت تبدیل N% همان اثر s% را دارند.

یادآوری ۷- در حال حاضر، هیچ بومی‌سازی به‌وسیله N% پشتیبانی نمی‌شود و قالب‌دهی حاصل از N% در شرایط بومی متفاوت تغییری نمی‌کند.

proc_exists()

وجود یک فرآیند را تایید می‌کند.

قالب

proc_exists(*pid*)

پارامتر

پارامتر	تعریف
pid	شماره شناسه فرآیندی که وجود آن باید مورد تأیید قرار گیرد.

توضیح

اگر فرآیند دارای شماره شناسه pid وجود داشته باشد، تابع ()pro_exists مقدار TRUE و اگر وجود نداشته باشد مقدار FALSE را برمی‌گرداند.

process()

فهرستی از فرآیندها را از حافظه نهان فرآیند عامل برمی‌گرداند.

قالب

process(*regular*_expression)

پارامتر

پارامتر	تعریف
regular_expression	دنباله کاراکتری که الگویی که تابع ()process باید در حافظه نهان فرآیند عامل به‌دنبال آن بگردد را تعریف می‌کند. پارامتر *regular_expression* با عبارات باقاعده‌ای که در فرمان ed(1) UNIX و توصیف UNIX regexp(5) تعریف شده است، مطابقت دارد. در زیر، خلاصه‌ای کوتاه از چند کاراکتر عبارت باقاعده ذکر می‌شود: ^ ابتدای سطر \< ابتدای یک کلمه $ انتهای سطر \> انتهای یک کلمه . با هر تک کاراکتری تطبیق داده می‌شود. * با صفر یا تعدادی تکرار آنچه پس از آن می‌آید تطبیق داده می‌شود. [] با هریک از کاراکترهای داخل ان تطبیق داده می‌شود. [^] با هر کاراکتری به‌جز کاراکترهای داخل آن تطبیق داده می‌شود.

توضیح

تابع ()process فهرستی از فرآیندهای موجود در حافظه نهان فرآیند عامل را که با عبارت باقاعده regular_expression تطبیق دارند، برمی‌گرداند. هر ورودی در این فهرست، رشته‌ای در قالب زیر است:

pid ppid user status size cputime *command_name command_line*

یادآوری ۸- برخی platformها همه مقادیر برگردانده شده را پشتیبانی نمی‌کنند. برای یک platform خاص، تابع ()process به‌طور عمومی همان اطلاعاتی را برمی‌گرداند که فرمان ps برمی‌گرداند. اگر هیچ فرآیندی با regular_expression مطابقت نداشته باشد، تابع ()process رشته NULL را برمی‌گرداند.

مثال

فرمان‌های SMSL زیر، همه شبح‌های فرآیند پایگاه داده ORACLE را فهرست می‌کنند.

```
#find ORACLE database daemons
ora_procs = process("ora_");
printf("%d", ora_procs);
```

پارامترها

پارامتر	تعریف
pid	شماره شناسه فرآیند
ppid	شماره شناسه فرآیند والد
user	نام کاربری که فرآیند متعلق به او است.
status	وضعیت فرآیند درون سامانه. بازه معتبر: ۰ غیرموجود S در حال خواب W در حال انتظار R در حال اجرا I میانی Z خاتمه‌یافته T متوقف‌شده X در حال رشد
size	اندازه تصویر هسته فرآیند (برحسب تعداد بلاک)
cputime	تعداد (عدد صحیح) ثانیه‌های CPU که به‌وسیله فرآیند مصرف شده است.
command_name	اولین کلمه خط فرمانی که فرآیند را شروع کرده است.
command_line	تمام خط فرمانی که فرآیند را شروع کرده است. به یاد داشته باشید که خط فرمان ممکن است در طی اجرای فرآیند تغییر کرده باشد.

random()

یک عدد تصادفی برمی‌گرداند.

قالب

random(*maximum*)

پارامتر

پارامتر	تعریف
maximum	بازه معتبر: maximum > 0 پیش‌فرض در صورتی که مشخص نشده باشد: $maximum = 2^{32} - 1$ (از تابع C زیرین)

توضیح

تابع ()random معادل تابع استاندارد ()random در زبان C استاندارد است.

اگر maximum صفر یا منفی باشد، تابع ()random یک پیغام خطای زمان اجرا را برمی‌گرداند. حد بالای اختیاری برای مقادیر برگردانده‌شده به‌وسیله random:

$$0 \leq random() \leq maximum - 1$$

read()

از یک فایل SMSL یا کانال فرآیند می‌خواند.

قالب

read(*channel, size*)

پارامترها

پارامتر	تعریف
channel	شماره کانال ورودی/خروجی فرآیند که تابع ()read، داده را از آن می‌خواند.
size	مقدار عدد صحیح که میزان داده‌ای که تابع ()read از کانال خواهد خواند را کنترل می‌کند. بازه معتبر: size > 0 به تابع ()read می‌گوید حداقل به تعداد size بایت خوانده و برگردد. size = 0 به تابع ()read می‌گوید به محض آنکه چیزی از کانال خواند، برگردد. size = -1 به تابع ()read می‌گوید تمام داده فراهم‌شده را از کانال خوانده و برگردد. پیش‌فرض در صورتی که مشخص نشده باشد: size = 0 است.

توضیح

تابع ()read داده‌هایی که از کانال خوانده است را برمی‌گرداند. در صورت مواجه با پایان فایل یا شرایط خطا، تابع ()read مقدار EOF (که عبارت است از رشته NULL) را برمی‌گرداند.

کانال‌ها با فراخوانی تابع ()fopen یا ()popen ایجاد می‌شوند.

یادآوری ۹- تابع ()read می‌تواند برای یک کانال فرآیند که با استفاده از تابع popen ایجاد شده است، مسدود شود، اما نه برای یک کانال فایل که با استفاده از تابع ()fopen ایجاد شده است.

به‌منظور تحمیل سریال‌سازی برای کانال‌های اشتراکی، هیچ دو فرآیند خواننده‌ای (یعنی توابع ()read یا ()readln) نمی‌توانند روی کانال یکسانی مسدود شوند. فرآیند خواننده دومی که تلاش کند روی کانال اشتراکی مسدود شود، با شکست مواجه خواهد شد و رشته NULL را برگردانده و متغیر شماره خطای SMSL را با E_SMSL_BUSY_CHANNEL مقداردهی می‌کند.

دیگر خرابی ممکن برای کانال اشتراکی می‌تواند با یک تابع ()close ایجاد شود که روی کانال که دارای یک فرآیند خواننده مسدودشده نیز هست، اجرا شده است. تابع ()close باعث می‌شود فرآیند خواننده رشته NULL را برگردانده و شماره خطای SMSL را با E_SMSL_UNBLOCKED_BY_CLOSE مقداردهی کند.

مثال

مثال SMSL زیر، یک کانال به سیستم عامل UNIX باز کرده، فرمان UNIX ls را اجرا کرده، سپس ورودی‌های شاخه که به‌وسیله فرمان ls بازگردانده شده‌اند را خوانده و چاپ می‌کند.

```
chan = popen("OS", "ls");
while((data = read(chan)) != EOF)
{
        printf("%s", data);
}
close(chan);
```

readln()

یک سطر داده را از یک فایل SMSL یا کانال فرآیند می‌خواند.

قالب

readln(*channel*)

پارامتر

پارامتر	تعریف
channel	شماره کانال ورودی/خروجی فرآیند را که تابع readln داده از آن می‌خواند.

توضیح

تابع ()readln سطر بعدی داده را از کانال خوانده و آن را برمی‌گرداند. در صورت مواجه با پایان فایل یا خطا، تابع ()readln مقدار EOF (NULL) را برمی‌گرداند.

کانال‌ها با فراخوانی تابع ()fopen یا ()popen ایجاد می‌شوند. به یاد داشته باشید که تابع ()readln می‌تواند برای یک کانال لوله که با استفاده از تابع ()popen ایجاد شده است مسدود شود، اما نه برای یک کانال فایل که با استفاده از تابع ()fopen ایجاد شده است.

به‌منظور تحمیل سریال‌سازی برای کانال‌های اشتراکی، هیچ دو فرآیند خواننده‌ای (یعنی توابع ()read یا ()readln) نمی‌توانند روی کانال یکسانی مسدود شوند. فرآیند خواننده دومی که تلاش کند روی کانال اشتراکی مسدود شود، با شکست مواجه خواهد شد، و رشته NULL را برگردانده و متغیر شماره خطای SMSL را با E_SMSL_BUSY_CHANNEL مقداردهی می‌کند.

دیگر خرابی ممکن برای کانال اشتراکی می‌تواند با یک تابع ()close ایجاد شود که روی کانالی که دارای یک فرآیند خواننده مسدودشده نیز هست، اجرا شده است. تابع ()close باعث می‌شود فرآیند خواننده رشته NULL را برگردانده و شماره خطای SMSL را با E_SMSL_UNBLOCKED_BY_CLOSE مقداردهی کند.

محدودیت

وقتی تابع ()readln روی فایل‌هایی که با تابع ()fopen باز شده‌اند اجرا شود، دارای محدودیت ۴ کیلوبایت برای هر سطر است . تابع ()readln ممکن است سطرهای طولانی‌تر از ۴ کیلوبایت را کوتاه کند. این محدودیت به کانال‌هایی که با استفاده از تابع ()popen باز شده‌اند اعمال نمی‌شود.

rindex()

آخرین وقوع یک رشته متنی را در دیگری برمی‌گرداند.

قالب

rindex(*text, string*)

پارامترها

پارامتر	تعریف
text	متنی که باید برای وقوع رشته مورد جستجو قرار گیرد. پارامتر text می‌تواند یک رشته متنی درون نماد نقل قول، یا یک یا چند فرمان SMSL باشد که متنی را به‌عنوان خروجی تولید می‌کنند.
string	یک یا چند کاراکتر درون نماد نقل قول که آخرین وقوع آن باید در داخل رشته مکان‌یابی شود.

توضیح

تابع ()rindex موقعیت آخرین وقوع رشته در داخل متن، یا در صورتی‌که رشته در متن وجود نداشته باشد مقدار صفر را برمی‌گرداند. موقعیت‌های درون رشته با شروع از یک، شماره‌گذاری می‌شوند.

set()

مقداری را به یک متغیر نسبت می‌دهد.

قالب

set(*variable, value*)

پارامترها

پارامتر	تعریف
variable	نام متغیری در سلسله‌مراتب شیء عامل که مقدار باید به آن نسبت داده شود.
value	مقدار عددی یا رشته‌ای که به متغیر نسبت داده می‌شود.

توضیح

تابع ()set مقدار متغیر را مقداردهی می‌کند. اگر متغیر یک نام وابسته باشد و در متن اسکریپت SMSL وجود نداشته باشد، تابع ()set به‌طور متوالی متن هر یک از نمونه‌های اولیه را جستجو می‌کند تا متغیر پیدا شود یا تا زمانی که جستجو در متن کامپیوتر با شکست مواجه شود.

اگر عمل انتساب موفق باشد، تابع ()set مقدار پارامتر value و در غیراین‌صورت رشته NULL را برمی‌گرداند.

مثال

دستور SMSL زیر، مقدار پارامتر MyParam مربوط به پایگاه داده Dev از RDB را با ۱۰ مقداردهی می‌کند.

set(“/RDB/Dev/MyParam/value”, 10);

share()

یک کانال محلی را به یک کانال سراسری اشتراکی تبدیل می‌کند.

قالب

share(*channel*, *name*)

پارامترها

پارامتر	تعریف
channel	شماره کانال ورودی/خروجی فرآیند که در زمان باز شدن کانال با استفاده از تابع fopen() یا popen()، برگردانده شده است.
name	رشته کاراکترای که برای شناسایی کانال اشتراکی در جدول کانال‌های سراسری استفاده می‌شود. توصیه می‌شود یک نام غیرعددی مشخص کنید تا از تصادم با شماره‌هایی که به‌طور داخلی برای کانال‌های محلی استفاده می‌شود، جلوگیری شود. استفاده از یک شماره برای نام عملاً موجب شکست تابع share() نمی‌شود، اما منجر به بروز یک اخطار زمان اجرای SMSL می‌شود. تابع share() به شکل وظیفه‌مندی نام شماره‌ای مشخص‌شده را در جدول سراسری قرار می‌دهد که منجر به تناقض‌های احتمالی با کانال‌های محلی در توابع close()، read()، write() و readln() می‌شود.

توضیح

تابع share() تابع اصلی برای استفاده از کانال‌های اشتراکی است . تابع share()کانال محلی موجود را به‌عنوان یک نام به جدول کانال‌های سراسری منتشر می‌کند. کانال‌های بازشده با هرکدام از توابع popen() یا fopen() می‌توانند به اشتراک گذاشته شوند.

اگر تابع share() موفق باشد، مقدار صفر را برمی‌گرداند. کانال محلی، دیگر در فرآیندی که آن را باز کرده است فراهم نبوده و نیازی به تابع close() ندارد. در واقع، تابع close() یک خطا باز خواهد گرداند، چرا که کانال محلی را پیدا نخواهد کرد.

تابع share() در شرایط زیر با شکست مواجه شده، مقدار ۱- را برگردانده و متغیر شماره خطای SMSL را مقداردهی می‌کند:

- کانال محلی وجود نداشته باشد؛
- نام کانال سراسری در حال حاضر در جدول کانال سراسری وجود داشته باشد.

در صورت شکست، کانال محلی تغییری نکرده و هنوز فراهم است . هیچ کانال سراسری اضافه نخواهد شد.

یک کانال سراسری، در زمان ارسال به توابع read()، readln()، write() و close() به‌وسیله نام مورد ارجاع قرار می‌گیرد. این توابع ابتدا جدول کانال محلی که تنها شامل شماره کانال‌ها است و سپس جدول کانال سراسری را جستجو می‌کنند.

sin()

سینوس آرگومان را برمی‌گرداند.

قالب

sin(*radians*)

پارامتر

پارامتر	تعریف
radians	طول (برحسب رادیان) کمانی که سینوس آن باید تعیین شود. بازه معتبر: $-\infty \leq \text{radians} \leq \infty$

توضیح

تابع sin() سینوس پارامتر radians را برمی‌گرداند. بازه خروجی برای تابع sin() برابر $-1 \leq \sin() \leq 1$ است .

sinh()

سینوس هایپربولیک آرگومان را برمی‌گرداند.

قالب

sinh(*argument*)

پارامتر

پارامتر	تعریف
argument	مقدار عددی که سینوس هایپربولیک آن باید تعیین شود. بازه معتبر: $-\infty \leq \text{argument} \leq \infty$

توضیح

تابع sinh() سینوس هایپربولیک آرگومان را برمی‌گرداند. سینوس هایپربولیک به‌وسیله عبارت زیر تعریف می‌شود:

$$\sinh(x) = (e^{x} - e^{-x})/2$$

که *e* پایه لگاریتم طبیعی است. ($e = 2.71828\ldots$) بازه خروجی برای تابع sinh() برابر $-\infty \leq \sinh() \leq \infty$ است .

sleep()

اجرای فرآیند را برای چند ثانیه به تعویق می‌اندازد.

قالب

sleep(*seconds*)

پارامتر

پارامتر	تعریف
seconds	عدد صحیحی که تعداد ثانیه‌هایی که فرآیند باید معلق بماند را مشخص می‌کند. بازه معتبر: $\text{seconds} > 0$ برابر تعداد ثانیه‌هایی است که فرآیند معلق خواهد ماند. $\text{seconds} \leq 0$ شمارنده بلافاصله منقضی می‌شود.

توضیح

تابع ()sleep یک فرآیند SMSL را برای تعداد ثانیه مشخص‌شده معلق می‌کند. در حالت تعلیق، فرآیند SMSL هیچ منبعی از CPU مصرف نکرده و تا زمانی‌که با انقضای شمارنده ثانیه‌ها بیدار نشود، تفسیر نخواهد شد.

یادآوری ۱۰- تابع ()sleep فقط فرآیندی که آن را فراخوانی می‌کند، معلق می‌کند. همه فرآیندهای SMSL دیگر به اجرای طبیعی خود ادامه می‌دهند.

اگر پارامتر seconds غیرعددی باشد، تابع ()sleep یک اخطار زمان اجرا را برمی‌گرداند که در این‌صورت شمارنده به مقدار پیش‌فرض صفر مقداردهی می‌شود.

sort()

فهرستی از مقادیر عددی یا الفبایی را مرتب می‌کند.

قالب

sort(*list, mode, position*)

پارامترها

پارامتر	تعریف
list	فهرست SMSL که عناصر آن باید مرتب شوند.
mode	رشته کاراکتری اختیاری که ترتیب مرتب‌سازی را مشخص می‌کند. رشته کاراکتری باید در داخل نماد نقل قول محصور شده باشد. بازه معتبر: "n" ترتیب عددی صعودی؛ "nr" ترتیب عددی نزولی؛ "" ترتیب الفبایی صعودی؛ "r" ترتیب الفبایی نزولی پیش‌فرض در صورتی که مشخص نشده باشد: "" (ترتیب الفبایی صعودی)
position	عدد صحیح اختیاری که موقعیت کاراکتر در داخل هر عنصر که مرتب‌سازی باید از آن موقعیت شروع شود را مشخص می‌کند. اولین کاراکتر هر عنصر فهرست، کاراکتر ۱ است . اگر طول هر عنصری در فهرست، کوچکتر از مقدار position باشد، اثر تابع مشابه حالتی است که همه عناصر فهرست عناصر NULL باشند. پارامتر position عناصر را کوتاه نمی‌کند؛ فقط به‌منظور مقایسه، position – 1 کاراکتر اول را نادیده می‌گیرد. پیش‌فرض در صورتی که مشخص نشده باشد: ۱ است.

توضیح

تابع ()sort نسخه مرتب‌شده‌ای از فهرست را که با توجه به مقدار پارامتر mode مرتب شده است، برمی‌گرداند.

تابع ()sort عناصر تکراری فهرست را ادغام نمی‌کند: فهرست بازگردانده‌شده به همان تعداد عناصر فهرست اولیه عنصر دارد. ترتیبی که عناصر تکراری طبق آن برگردانده می‌شوند، تعریف نشده است زیرا برای تابع کتابخانه‌ای ()qsort در زبان C تعریف نشده است. این واقعیت در موارد زیر مرتبط است :

- مرتب‌سازی عددی رشته‌های دارای مقادیر پیشوند عددی یکسان اما پسوندهای غیرعددی متفاوت؛
- هر حالت مرتب‌سازی که در آن مقدار پارامتر position از بیش از یک عنصر فهرست بزرگتر است. (تابع sort() همه‌ی چنین عناصری را به‌عنوان عناصر NULL تکراری به‌حساب می‌آورد.)

اگر فهرست، فهرست NULL باشد، تابع sort() فهرست NULL را برمی‌گرداند. برای یک فهرست غیرتهی، تابع sort() همیشه یک فهرست خوش-تعریف که آخرین سطر آن به‌طور مناسب با یک کاراکتر سطر جدیدخاتمه یافته، را برمی‌گرداند.

یادآوری ۱۱- نیازی نیست فهرست با یک کاراکتر سطر جدید خاتمه یافته باشد. مرتب‌سازی عددی براساس مقادیر ممیز شناور انجام می‌شود؛ ورودی‌های غیرعددی فهرست، طبق تابع atof() از کتابخانه C استاندارد موجود در سامانه تبدیل می‌شوند.

sprintf()

قالب مشخص شده را به‌عنوان یک رشته کاراکتری به یک مقصد برمی‌گرداند.

قالب

sprintf(*format*)

پارامتر

پارامتر	تعریف
format	متن، نام متغیرها و کاراکترهای کنترلی که محتوا و قالب خروجی به کامپیوتر یا پنجره خروجی کار را مشخص می‌کند. پارامتر format اجازه استفاده از کاراکترهای تبدیل زیر را فراهم می‌کند: %d عدد در مبنای ده علامت‌دار (مشابه %i) %i عدد در مبنای ده علامت‌دار (مشابه %d) %u عدد در مبنای ده بدون علامت %o عدد در مبنای هشت بدون علامت %x عدد در مبنای شانزده بدون علامت با استفاده از abcdef %X عدد در مبنای شانزده بدون علامت با استفاده از ABCDEF %c کاراکتر بدون علامت %s رشته کاراکتری %e عدد اعشاری دقت مضاعف به شکل drdde±ddd که هر d یک رقم و r کاراکتر مبنا است. %E عدد اعشاری دقت مضاعف به شکل drdddE±ddd که هر d یک رقم و r کاراکتر مبنا است. %f عدد در مبنای ده به شکل dddrddd که هر d یک رقم و r کاراکتر مبنا است. %g اگر توان بعد از تبدیل کمتر از ۴- است، به شیوه %e و در غیراین‌صورت به شیوه %f چاپ می‌کند. %G به شیوه %E و با درنظر گرفتن اینکه دقت، تعداد ارقام با ارزش را مشخص می‌کند، چاپ می‌کند. %N ارقام را در گروه‌های سه‌تایی گروه‌بندی کرده و آنها را با ویرگول‌هایی که در سمت راست رشته آغاز می‌شوند جدا می‌کند. %% یک کاراکتر % را چاپ می‌کند.

	پارامتر format از کاراکترهای تبدیل اشاره‌گرهای C، یعنی p% و n% پشتیبانی نمی‌کند. پارامتر format اجازه استفاده از پرچم‌های زیر را می‌دهد: - چپ چین کردن و پرکردن سمت راست با فضاهای خالی + نمایش علامت به‌علاوه وقتی مقدار بزرگتر از صفر است. ۰ پرکردن با استفاده از صفر، اگر هیچ نحوه پرکردن دیگری مشخص نشده باشد. # معنای یک تبدیل را تغییر می‌دهد: 0x یا 0X را به تبدیل‌های x% و X% اضافه می‌کند. همیشه کاراکتر مبنا را به تبدیل‌های e%، E%، f%، g% و G% اضافه می‌کند. صفرهای انتهایی را در تبدیل‌های g% و G% حفظ می‌کند. پرچم # بر تبدیل‌های c%، d%، s% یا i% اثری ندارد.

توضیح

تابع sprintf() مانند تابع printf() است به‌جز این‌که به‌جای چاپ رشته ایجاد شده در خروجی، آن‌را برمی‌گرداند. اگر پارامتر format خطایی داشته باشد، sprintf متغیر شماره خطای SMSL را مقداردهی کرده و رشته NULL را برمی‌گرداند.

قالب‌ها، تبدیل‌ها و مقادیر شماره خطا برای خطاهای مختلف، مشابه آن چیزی است که برای تابع printf() توضیح داده شد.

برنامه‌نویسان C باید مراقب باشند تا به‌جای استفاده از شیوه C:

sprintf(*destination*, *format*)

از شیوه SMSL استفاده کنند:

destination = sprintf(*format*)

شیوه C اغلب منجر به یک اخطار زمان ترجمه درباره دستور بی‌اثر می‌شود.

قراردادهای C که به‌وسیله تابع sprintf زبان SMSL پشتیبانی نمی‌شوند

تابع sprintf() قرارداد C مبنی بر استفاده از ستاره (*) به‌عنوان طول فیلد یا نشانگر دقت را پشتیبانی نمی‌کند. تابع sprintf کاراکترهای تبدیل p% و n% را پشتیبانی نمی‌کند.

اصلاح‌کننده‌های طول h، l و L معتبر نیستند و تابع sprintf() آن‌ها را نادیده می‌گیرد.

تبدیل قالب‌های تابع sprintf() بر روی هر platform، به‌طور مستقیم به رویه کتابخانه‌ای sprintf() در زبان C ارسال می‌شود. خروجی برای ویژگی‌های قالب‌دهی مبهم ممکن است بین platformها متفاوت باشد.

تفاوت‌های تبدیل بین رویه sprintf کتابخانه C و تابع sprintf زبان SMSL

تبدیل‌های قالب، بین C استاندارد و SMSL، معنای یکسانی دارند، اما مفهوم انواع متغیر بین آن دو تفاوت دارد.

زبان SMSL فقط انواع رشته‌ای را برای متغیرهایش پشتیبانی می‌کند و بنابراین آرگومان‌های رشته‌ای تابع sprintf() طبق روش مناسبی برای تبدیل قالب، تبدیل می‌شوند:

- قالب‌های عدد صحیح مانند d% رشته را به اعداد صحیح علامت‌دار تبدیل می‌کنند.
- قالب‌های عددی غیر عدد صحیح مانند f% به مقادیر ممیز شناور تبدیل می‌کنند.

- %c معادل ASCII آرگومان عددی‌اش، یا اولین کاراکتر آرگومانش را برای آرگومان‌های غیرعددی چاپ می‌کند. (اعمال %c به "65" موجب چاپ 'A' و اعمال %c به "AB" موجب چاپ 'A' خواهد شد.)
- %s منجر به هیچ تبدیلی نمی‌شود.
- %% نیازی به آرگومان ندارد.

sqrt()

ریشه دوم (جذر) آرگومان را برمی‌گرداند.

قالب

sqrt(*argument*)

پارامتر

پارامتر	تعریف
argument	مقدار عددی که ریشه دوم ریاضی آن باید برگردانده شود. بازه معتبر: $-\infty \leq \text{argument} \leq \infty$

توضیح

تابع sqrt() ریشه دوم آرگومان عدد صحیح مثبت یا عدد حقیقی را برمی‌گرداند.

srandom()

مولد اعداد تصادفی را با یک مقدار بذر مقداردهی اولیه می‌کند.

قالب

srandom(*seed*)

پارامتر

پارامتر	تعریف
seed	مقدار عددی که به‌عنوان نقطه شروع برای تولید اعداد شبه‌تصادفی به‌وسیله تابع random() مورد استفاده قرار می‌گیرد.

توضیح

تابع srandom() بذر عدد تصادفی را برای تابع random() مقداردهی می‌کند. پارامتر *seed* به‌طور مستقیم به تابع UNIX srandom()C ارسال می‌شود.

تابع srandom() زبان SMSL همیشه رشته NULL را برمی‌گرداند.

subset()

اینکه آیا یک فهرست SMSL، زیرمجموعه‌ای از دیگری است را تأیید می‌کند.

قالب

subset(*set, subset*)

پارامترها

پارامتر	تعریف
set	فهرست SMSL، یعنی حکم مجموعه را در تأیید مجموعه-زیرمجموعه دارد.
subset	فهرست SMSL، یعنی حکم زیرمجموعه را در تأیید مجموعه-زیرمجموعه دارد.

توضیح

تابع ()subset یک مقدار بولی ۰ یا ۱ را برمی‌گرداند که نشان می‌دهد آیا subset یک زیرمجموعه مناسب یا نامناسب از set است یا خیر. اگر subset مجموعه NULL باشد، تابع ()subset مقدار ۱ (TRUE) را برمی‌گرداند. اگر set مجموعه NULL باشد و subset نباشد، تابع ()subset مقدار صفر (FALSE) را برمی‌گرداند.

تابع ()subset عناصر تکراری را نادیده گرفته و فقط اگر همه عناصر subset در set هم حضور داشته باشند، مقدار ۱ را برمی‌گرداند.

مثال

تابع ()subset می‌تواند برای تعیین اینکه آیا عنصر خاصی، در یک مجموعه حضور دارد یا خیر استفاده شود و بنابراین کارکرد "is_member" را مانند زیر فراهم می‌کند:

```
if (subset(my_set, "blue"))
{
        # SMSL set "my_set" contains element "blue"
}
```

قرار دادن سطر جدید در انتهای رشته "blue" ضروری نیست، چرا که به‌وسیله تابع ()subset درج می‌شود. با دستورات مثال بالا، به‌عنوان یک تابع ()subset که روی یک مجموعه با یک عنصر عمل می‌کند، برخورد می‌شود.

substr()

بخش مشخصی را از یک رشته از کاراکترها برمی‌گرداند.

قالب

substr(*text, start, length*)

پارامترها

پارامتر	تعریف
text	متنی که یک زیررشته از کاراکترها باید از آن برگردانده شود. پارامتر *text* می‌تواند یک رشته متنی که در علامت نقل قول قرار گرفته، یا یک یا چند فرمان SMSL باشد که به‌عنوان خروجی یک متن تولید می‌کنند.
start	موقعیت کاراکتری در متن که باید اولین کاراکتر زیررشته باشد. کاراکتر اول در متن، کاراکتر موقعیت ۱ است .
length	تعداد کل کاراکترهایی از متن که باید در زیررشته برگردانده شود.

توضیح

تابع ()substr زیررشته‌ای از متن با طول length کاراکتر را که از موقعیت start شروع می‌شود، برمی‌گرداند.

system()

یک فرمان را به سیستم عامل کامپیوتر ارائه می‌کند.

قالب

system(*command, instance*)

پارامترها

پارامتر	تعریف
command	نحو فرمان سیستم عامل ارائه شده. پارامتر *command* می‌تواند شامل بازهدایت خروجی، لوله‌ها، wildcard و از این قبیل باشد.
instance	نمونه برنامه کاربردی که فرمان باید در برابر آن اجرا شود. مقدار پیش‌فرض در صورتی که مشخص نشده باشد: نمونه برنامه کاربردی که نزدیکترین نمونه اولیه‌ی فرمان است.

توضیح

تابع ()system هر خروجی را که به‌وسیله فرمان ارائه‌شده تولید شود، به زیرسامانه اجرای فرمان وابسته به سامانه برمی‌گرداند.

tail()

آخرین سطرهای یک بلوک متنی را برمی‌گرداند.

قالب

tail(*text, lines*)

پارامترها

پارامتر	تعریف
text	متنی که آخرین سطرهای آن باید به‌وسیله ()tail برگردانده شود. پارامتر *text* می‌تواند یک رشته متنی که در علامت نقل قول قرار گرفته، یا یک یا چند فرمان SMSL باشد که به‌عنوان خروجی یک متن تولید می‌کنند.
lines	تعداد سطرهایی از متن، با شروع از آخرین سطر متن که باید برگردانده شود.

توضیح

تابع ()tail آخرین lines سطر متن را برمی‌گرداند.

tan()

تانژانت آرگومان را برمی‌گرداند.

قالب

tan(*radians*)

پارامتر

پارامتر	تعریف
radians	طول (برحسب رادیان) کمانی که تانژانت آن باید تعیین شود. بازه معتبر: $-\infty \leq radians \leq \infty$

توضیح

تابع tan() تانژانت پارامتر radians را برمی‌گرداند. بازه خروجی برای تابع tan() برابر $-\infty < tan() < \infty$ است. تابع tan() وقتی $radians = p(2n+1)/2$ که n عددی صحیح است، تعریف نشده است.

tanh()

تانژانت هایپربولیک آرگومان را برمی‌گرداند.

قالب

tanh(*argument*)

پارامتر

پارامتر	تعریف
argument	مقدار عددی که تانژانت هایپربولیک آن باید تعیین شود. بازه معتبر: $-\infty \leq argument \leq \infty$

توضیح

تابع tanh() تانژانت هایپربولیک آرگومان را برمی‌گرداند. تانژانت هایپربولیک به‌وسیله عبارت زیر تعریف می‌شود:

$$\tanh(x) = (e^{x} - e^{-x}) / (e^{x} - e^{-x})$$

که e پایه لگاریتم طبیعی ($e = 2.71828\ldots$) است. بازه خروجی برای تابع tanh() برابر $-1 \leq tanh() \leq 1$ است.

time()

تعداد ثانیه‌های سپری‌شده از اول ژانویه ۱۹۷۰، ساعت 00:00:00 GMT را برمی‌گرداند.

قالب

time()

توضیح

تابع time() زمان جاری را در قالب تعداد ثانیه‌هایی که از ساعت 00:00:00 GMT اول ژانویه ۱۹۷۰ سپری شده است، برمی‌گرداند.

Tmpnam()

نامی یکتا را برای ایجاد فایل موقت برمی‌گرداند.

قالب

tmpnam()

توضیح

تابع ()tmpnam نامی را برمی‌گرداند که یکتا بودن آن تضمین شده است و می‌تواند برای ارسال به تابع fopen به‌منظور ایجاد فایل‌های موقت استفاده شود.

معنای تابع ()tmpnam مشابه رویه ()tmpnam در C است - به‌ویژه تعداد محدودی نام یکتا، چنانکه به‌وسیله ثابت TMP_MAX زبان C تعریف شده است، به‌وسیله رویه ()tmpnam برگردانده می‌شود. همه فرآیندهای SMSL روی یک عامل داده شده، مجموعه یکسانی از نام‌ها را به اشتراک می‌گذارند و ممکن است خطر تداخل نام‌ها پیش آید. اگر اندازه TMP_MAX مسأله باشد، پسوندی به نام فایل برگردانده‌شده را بیافزایید.

مثال

مثال زیر چگونگی استفاده از تابع ()tmpnam برای تولید یک نام فایل موقت را نشان می‌دهد. تابع SMSL پسوندی به نام برگردانده‌شده را اضافه می‌کند تا یکتا بودن آن را تضمین کند.

name = tmpnam() . “.dave”; fp = fopen(name, “w”);

tolower()

تمام متن را به کاراکترهای حروف کوچک تبدیل می‌کند.

قالب

tolower(*text*)

پارامتر

پارامتر	تعریف
text	متنی که باید در قالب حروف کوچک برگردانده شود. پارامتر *text* می‌تواند یک رشته متنی که در علامت نقل قول قرار گرفته، یا یک یا چند فرمان SMSL باشد که به‌عنوان خروجی یک متن تولید می‌کنند.

توضیح

تابع ()tolower یک کپی از متن که در آن همه حروف بزرگ به حروف کوچک تبدیل شده است را برمی‌گرداند.

toupper()

تمام متن را به کاراکترهای حروف بزرگ تبدیل می‌کند.

قالب

toupper(*text*)

پارامتر

پارامتر	تعریف
text	متنی که باید در قالب حروف بزرگ برگردانده شود. پارامتر *text* می‌تواند یک رشته متنی که در علامت نقل قول قرار گرفته، یا یک یا چند فرمان SMSL باشد که به‌عنوان خروجی یک متن تولید می‌کنند.

توضیح

تابع ()toupper یک کپی از متن که در آن همه حروف کوچک به حروف بزرگ تبدیل شده است را برمی‌گرداند.

trim()

کاراکترهای ناخواسته را از متن حذف می‌کند.

قالب

trim(*text, unwanted*)

پارامترها

پارامتر	تعریف
text	متنی که باید بدون کاراکترهای مشخص شده برگردانده شود. پارامتر *text* می‌تواند یک رشته متنی که در علامت نقل قول قرار گرفته، یا یک یا چند فرمان SMSL باشد که به‌عنوان خروجی یک متن تولید می‌کنند.
unwanted	یک یا تعداد بیشتری کاراکتر که باید به‌وسیله تابع trim از کپی خروجی متن حذف شوند.

توضیح

تابع ()trim یک کپی از متن را برمی‌گرداند که همه وقوع‌های کاراکترهای موجود در پارامتر unwanted از آن حذف شده است.

union()

فهرستی را برمی‌گرداند که اجتماع فهرست‌های مجزایی است.

قالب

union(*list1, list2, list3, list4…listn*)

پارامترها

پارامتر	تعریف
listn	فهرست SMSL شامل عناصری که باید با هم اجتماع گرفته شوند و در یک فهرست خوش-تعریف تنها برگردانده شوند. فقط دو فهرست ورودی اول، list1 و list2، ضروری هستند؛ بقیه فهرست‌ها اختیاری هستند.

توضیح

تابع ()union یک فهرست SMSL را برمی‌گرداند که عناصر حاصل از ادغام همه فهرست‌های پارامتر را شامل می‌شود. برخلاف توابع ()difference و ()intersection، فهرست برگردانده‌شده به‌وسیله تابع union یک مجموعه خوش-تعریف بدون هیچ عنصر تکراری است . تابع ()union یک سطر جدید به انتهای هر فهرست غیرتهی که فاقد آن است اضافه می‌کند. اگر مقدار بازگشتی، فهرست NULL نباشد، مجموعه بازگردانده‌شده همیشه به یک سطر جدید ختم می‌شود تا به این ترتیب همه عناصر مجموعه با یک کاراکتر سطر جدید تمام شوند.

unique()

عناصر تکراری را از یک فهرست حذف می‌کند.

قالب

unique(*list*)

پارامتر

پارامتر	تعریف
list	فهرست SMSL شامل عناصری که باید در یک فهرست خوش-تعریف (یکتا) تنها برگردانده شوند.

توضیح

تابع unique() یک فهرست SMSL خوش-تعریف که همه عناصر تکراری آن حذف شده‌اند را برمی‌گرداند. همه عناصری که در مقدار بازگشتی باقی می‌مانند، به همان ترتیبی که در فهرست بوده‌اند ظاهر می‌شوند. اگر فهرست یک فهرست NULL باشد، تابع unique فهرست NULL را برمی‌گرداند؛ در غیراین‌صورت تابع unique() فهرستی را برمی‌گرداند که با یک کاراکتر سطر جدید تمام شده است، به این ترتیب همه عناصر فهرست موجود در فهرست به یک کاراکتر سطر جدید ختم می‌شوند.

unlock()

یک قفل فرآیند SMSL را رها می‌کند.

قالب

unlock(*lockname*)

پارامتر

پارامتر	تعریف
lockname	نام قفلی که باید رها شود.

توضیح

تابع unlock() قفل *lockname* را که با فراخوانی پیشین تابع lock()به این فرآیند تخصیص داده شده بود ، رها می‌کند. تابع unlock() در صورت موفقیت مقدار ۱ و در صورت شکست مقدار صفر را برمی‌گرداند. اگر هیچ قفلی به نام *lockname* وجود نداشته باشد یا در حال حاضر تحت مالکیت این فرآیند نباشد، تابع unlock() یک خطای زمان اجرا را گزارش کرده، متغیر شماره خطای SMSL را مقداردهی کرده و مقدار صفر را برمی‌گرداند.

یادآوری ۱۲- همه قفل‌های نگه‌داشته‌شده به‌وسیله یک فرآیند، وقتی فرآیند خارج می‌شود به روشی مشابه اجرای تابع unlock، به‌طور خودکار رها می‌شوند. توصیه می‌شود قفل‌ها را به‌جای آنکه به‌طور ضمنی با استفاده از خروج فرآیند رها کنید، به‌طور صریح با استفاده از تابع unlock() رها کنید. اگر این فرآیند تنها فرآیندی است که قفل را نگه‌داشته و فرآیندهایی برای آن در صف قرار گرفته-اند، نخستین فرآیند منتظر بیدار شده و مجوز استفاده از قفل را دریافت می‌کند. اگر اولین فرآیند، یک درخواست اشتراکی است، آنگاه هر فرآیند دیگری که برای یک قفل اشتراکی در صف قرار گرفته است نیز مجوز دسترسی اشتراکی به قفل را دریافت می‌کند. (به‌جز برای فرآیندهایی که در صف مربوط به این قفل، در پشت یک درخواست نویسنده قرار دارند.)

write()

در یک فرآیند SMSL یا کانال فایل می‌نویسد.

قالب

write(*chan*,*text*)

پارامترها

پارامتر	تعریف
chan	شماره کانال ورودی/خروجی فرآیند که *text* باید در آن نوشته شود.
text	متنی که باید در کانال *chan* نوشته شود. پارامتر *text* می‌تواند یک رشته متنی که در علامت نقل قول قرار گرفته، یا یک یا چند دستور SMSL باشد که به‌عنوان خروجی یک متن تولید می‌کنند.

توضیح

تابع ()write متن را در کانال *chan* می‌نویسد. تابع ()write تعداد کاراکترهای نوشته شده، یا در صورت بروز خطا مقدار ۱- را برمی‌گرداند.

اگر امکان نوشتن بلافاصله متن وجود نداشته باشد، فراخوانی تابع ()write تا زمانی که بتواند همه متن را بنویسد یا کانال خاتمه پیدا کند، مسدود می‌شود.

یادآوری ۱۳- تابع ()write می‌تواند برای یک کانال فرآیند که با استفاده از تابع ()popen ایجاد شده است مسدود شود، اما نه برای یک کانال فایل که با استفاده از تابع ()fopen ایجاد شده است.

به‌منظور تحمیل سریال‌سازی برای کانال‌های اشتراکی، هیچ یک از دو فرآیند خواننده (یعنی توابع ()read یا ()readln) نمی‌توانند روی کانال یکسانی مسدود شوند. دومین فرآیند خواننده که تلاش کند روی کانال اشتراکی مسدود شود، با شکست مواجه خواهد شد و رشته NULL را برگردانده و متغیر شماره خطای SMSL را با E_SMSL_BUSY_CHANNEL مقداردهی می‌کند.

دیگر خرابی ممکن برای کانال اشتراکی می‌تواند با یک تابع ()close ایجاد شود که روی کانالی که دارای یک فرآیند خواننده مسدودشده نیز هست، اجرا شده است. تابع ()close باعث می‌شود فرآیند خواننده رشته NULL را برگردانده و شماره خطای SMSL را با E_SMSL_UNBLOCKED_BY_CLOSE مقداردهی کند.

پیوست ح
(الزامی)
پیش‌نویس MOCS

این پیوست شامل پیش‌نویس MOCS برای زیرمجموعه‌ای از کلاس‌های شیء تعریف‌شده در [X721] است که برای مدیریت SDH استفاده می‌شود.

نشانه‌گذاری‌های رایج زیر که در توصیه‌نامه X.724 تعریف شده‌اند برای ستون‌های وضعیت استفاده می‌شوند:

m اجباری

o اختیاری

c شرطی

x ممنوع

- غیرقابل اجرا یا خارج از محدوده

به یاد داشته باشید که "c"، "m"، "o" و "x" وقتی به‌صورت تودرتو در زیر یک آیتم شرطی یا اختیاری از جدول یکسانی قرار گیرند دارای پیشوند "c:" هستند.

در ستون وضعیت، نیازمندی‌های ایستا به شکل زیر بیان می‌شوند:

m برای مشخصه‌هایی که در بسته‌های اجباری یا در بسته‌های شرطی قرار دارند، اگر شرط GDMO همیشه درست باشد.

o برای مشخصه‌های بسته‌های شرطی با شرط‌های GDMO که اختیاری بودن ایستا را نشان می‌دهند، برای مثال «اگر یک نمونه آن را پشتیبان می‌کند.»

cn برای تمامی شرط‌های دیگر، جایی که "n" یک عدد صحیح یکتا و "cn" یک ارجاع به عبارت وضعیت شرطی ، آن‌طوری که در ITU-T Rec. X.291 | ISO/IEC 9646-2 و ITU-T Rec. X.296 | ISO/IEC 9646-7 تعریف شده است. هر شرط نشانه‌گذاری‌شده با "cn"، با جدول دربرگیرنده‌اش رابطه نسبی دارد.

x برای مشخصه‌هایی که به‌صورت صریح با تعریف ممنوع شده‌اند.

- برای مشخصه‌هایی که با تعریف به آنها اشاره‌ای نشده است.

نشانه‌گذاری‌های رایج زیر که در ITU-T Rec. X.724 | ISO/IEC 10165-6 و Rec. X.296 | ISO/IEC 9646-7 تعریف شده‌اند برای ستون‌های پشتیبانی جواب استفاده می‌شوند:

Y پیاده‌سازی‌شده

N پیاده‌سازی‌نشده

- نیازی به جواب ندارد.

کوته‌نوشت‌های زیر مورد استفاده قرار می‌گیرند:

```
smi2AttributeID      { joint-iso-itu-t ms(9) smi(3) part2(2) attribute(7) }
smi2MObjectClass     { joint-iso-itu-t ms(9) smi(3) part2(2) managedObjectClass(3) }
smi2Notification     { joint-iso-itu-t ms(9) smi(3) part2(2) notification(10) }
```

ح-۱ بیانیه انطباق با کلاس شیء basicSpawnerClass

جدول ح-۱- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی همه ویژگی‌های اجباری	آیا کلاس واقعی مانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	basicSpawnerClass	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx1(1)}		

اگر پاسخ پرسش کلاس واقعی در پشتیبانی کلاس شیء مدیریت‌شده در جدول ح-۱ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۲ تکمیل کند.

جدول ح-۲- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت‌شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۱-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۳ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۳- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-1/1b) then m else –						

ح-۱-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۴ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی

می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید برای هر یک از عملیات‌ها و برای هر یک از صفات پشتیبانی‌شده، پشتیبانی نشان دهد.

جدول ح-۴- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
۱	allomorphs	{smi2Attribute ID 50}		x		c1		x	
۲	nameBinding	{smi2Attribute ID 63}		–		m		x	
۳	objectClass	{smi2Attribute ID 65}		–		m		x	
۴	packages	{smi2Attribute ID 66}		–		m		x	

جدول ح-۴- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
c1: if not (H-1/1b) then m else –							

ح-۱-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۱-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۱-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۱-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۲ بیانیه انطباق با کلاس شیء commandSequencer

جدول ح-۵- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی همه ویژگی‌های اجباری	آیا کلاس واقعی مانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	commandSequencer	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx2(2)}		

اگر پاسخ پرسش کلاس واقعی در پشتیبانی کلاس شیء مدیریت‌شده در جدول ح-۵ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۶ تکمیل کند.

جدول ح-۶- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت‌شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۲-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۷ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۷- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		c2		
c1: if not (H-5/1b) then m else – c2: if H-7/1 then m else –						

ح-۲-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۸ بیان کند آیا صفات مشخص شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید برای هر یک از عملیات‌ها و برای هر یک از صفات پشتیبانی‌شده، پشتیبانی نشان دهد.

جدول ح-۸- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
۱	administrativeState	{smi2AttributeID 31}		m		m		m	
۲	allomorphs	{smi2AttributeID 50}		x		c1		x	
۳	commandSequencerId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx2(2)}		–		m		x	
۴	nameBinding	{smi2AttributeID 63}		–		m		x	
۵	objectClass	{smi2AttributeID 65}		–		m		x	
۶	operationalState	{smi2AttributeID 35}		–		m		x	
۷	packages	{smi2AttributeID 66}		–		c2		x	

جدول ح-۸- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
۵	x		x		x		
۶	x		x		x		
۷	x		x		x		

c1: if not (H-5/1b) then m else –
c2: if H-7/2 then m else –

ح-۲-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۲-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۲-۵ اعلان‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» از جدول ح-۹ بیان کند که آیا اعلان‌های مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی را در قالب مدهای تأییدشده و تأییدنشده نشان دهد.

جدول ح-۹- MOCS – پشتیبانی از اعلان

ردیف	برچسب قالب نوع اعلان	مقدار شناسه شیء برای نوع اعلان	قیود و مقادیر	وضعیت	پشتیبانی		اطلاعات افزوده
					تأیید شده	تأیید نشده	
۱	objectCreation }	{smi2Notification 6		m			
۲	objectDeletion	{smi2Notification 7}		m			
۳	stateChange	{smi2Notification 14}		m			

جدول ح-۹- (نهایی)

ردیف	زیرردیف	برچسب نام فیلد اعلان	مقدار شناسه شیء نوع صفت مرتبط با فیلد	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	1.1	additionalInformation	{smi2AttributeID 6}		o		
	1.1.1	identifier	–		c:m		
	1.1.2	significance	–		c:m		
	1.1.3	information	–		c:m		
	1.2	additionalText	{smi2AttributeID 7}		o		
	1.3	attributeList	{smi2AttributeID 9}		o		
	1.3.1	attributeId	–		c:m		
	1.3.1.1	globalForm	–		c:o.1		
	1.3.1.2	localForm	–		c:o.1		
	1.3.2	attributeValue	–		c:m		
	1.4	correlatedNotifications	{smi2AttributeID 12}		o		
	1.4.1	correlatedNotifications	–		c:m		
	1.4.2	sourceObjectInst	–		c:o		
	1.4.2. 1	distinguishedName	–		c:o.2		
	1.4.2.1.1	AttributeType	–		c:m		
	1.4.2.1.2	AttributeValue	–		c:m		
	1.4.2.2	nonSpecificForm	–		c:o.2		
	1.4.2.3	localDistinguishedName	–		c:o.2		
	1.4.2.3.1	AttributeType	–		c:m		

		c:m		–	AttributeValue	1.4.2.3.2	
		o		{smi2AttributeID 16}	notificationIdentifier	1.5	
		o		{smi2AttributeID 26}	sourceIndicator	1.6	
		o		{smi2AttributeID 6}	additionalInformation	2.1	2
		c:m		–	identifier	2.1.1	
		c:m		–	significance	2.1.2	
		c:m		–	information	2.1.3	
		o		{smi2AttributeID 7}	additionalText	2.2	
		o		{smi2AttributeID 9}	attributeList	2.3	
		c:m		–	attributeId	2.3.1	
		c:o.3		–	globalForm	2.3.1.1	
		c:o.3		–	localForm	2.3.1.2	
		c:m		–	attributeValue	2.3.2	
		o		{smi2AttributeID 12}	correlatedNotifications	2.4	
		c:m		–	correlatedNotifications	2.4.1	
		c:o		–	sourceObjectInst	2.4.2	
		c:o.4		–	distinguishedName	2.4.2.1	
		c:m		–	AttributeType	2.4.2.1.1	
		c:m		–	AttributeValue	2.4.2.1.2	
		c:o.4		–	nonSpecificForm	2.4.2.2	
		c:o.4		–	localDistinguishedName	2.4.2.3	
		c:m		–	AttributeType	2.4.2.3.1	
		c:m		–	AttributeValue	2.4.2.3.2	
		o		{smi2AttributeID 16}	notificationIdentifier	2.5	
		o		{smi2AttributeID 26}	sourceIndicator	2.6	
		o		{smi2AttributeID 6}	additionalInformation	3.1	3
		c:m		–	identifier	3.1.1	
		c:m		–	significance	3.1.2	
		c:m		–	information	3.1.3	
		o		{smi2AttributeID 7}	additionalText	3.2	
		o		{smi2AttributeID 8}	attributeIdentifierList	3.3	
		c:o.5		–	globalForm	3.3.1	
		c:o.5		–	localForm	3.3.2	
		o		{smi2AttributeID 12}	correlatedNotifications	3.4	

		c:m		–	correlatedNotifications	3.4.1	
		c:o		–	sourceObjectInst	3.4.2	
		c:o.6		–	distinguishedName	3.4.2.1	
		c:m		–	AttributeType	3.4.2.1.1	
		c:m		–	AttributeValue	3.4.2.1.2	
		c:o.6		–	nonSpecificForm	3.4.2.2	
		c:o.6		–	localDistinguishedName	3.4.2.3	
		c:m		–	AttributeType	3.4.2.3.1	
		c:m		–	AttributeValue	3.4.2.3.2	
		o		{smi2AttributeID 16}	notificationIdentifier	3.5	
		o		{smi2AttributeID 26}	sourceIndicator	3.6	
		m		{smi2AttributeID 28}	stateChangeDefinition	3.7	
		m		–	attributeID	3.7.1	
		c:o.7		–	globalForm	3.7.1.1	
		c:o.7		–	localForm	3.7.1.2	
		o		–	oldAttributeValue	3.7.2	
		m		–	newAttributeValue	3.7.3	

ح-۲-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۳ بیانیه انطباق با کلاس شیء generalStringScript

جدول ح-۱۰- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	generalStringScript	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx3(3)}		

اگر پاسخ پرسش کلاس واقعی در پشتیبانی کلاس شیء مدیریت‌شده در جدول ح-۱۰ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۱۱ تکمیل کند.

جدول ح-۱۱- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۳-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۱۲ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۱۲- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		c2		
c1: if not (H-10/1b) then m else –						

ح-۳-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۱۳ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۱۳- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
۱	administrativeState	{smi2Attribute ID 31}		m		m		m	
۲	allomorphs	{smi2Attribute ID 50}		x		c1		x	
۳	executionResultType	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx2(2)}		–		m		x	
۴	nameBinding	{smi2Attribute ID 63}		–		m		x	
۵	objectClass	{smi2Attribute ID 65}		–		m		x	
۶	packages	{smi2Attribute ID 66}		–		c2		x	

	m		m		m		{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx4(4)}	scriptContent	۷
	x		m		–		{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx5(5)}	scriptId	۸
	m		m		m		{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx7(7)}	scriptLanguageName	۹

جدول ح-۱۳- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	**وضعیت**	**پشتیبانی**	**وضعیت**	**پشتیبانی**	**وضعیت**	**پشتیبانی**	**اطلاعات افزوده**
1	x		x		x		
2	x		x		x		
3	x		x		x		
4	x		x		x		
5	x		x		x		
6	x		x		x		
7	x		x		x		
8	x		x		x		
9	x		x		x		
c1: if not (H-10/1b) then m else –							

ح-۳-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۳-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۳-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۳-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۴ بیانیه انطباق با کلاس شیء launchPad

جدول ح-۱۴- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت شده ای که ادعای انطباق با آن صورت گرفته است می‌باشد؟ (Y/N)
۱	launchPad	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx6(6)}		

اگر پاسخ پرسش کلاس واقعی در پشتیبانی کلاس شیء مدیریت‌شده در جدول ح-۱۴ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۱۵ تکمیل کند.

جدول ح-۱۵- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۴-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۱۶ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۱۶- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-14/1b) then m else –						

ح-۴-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۱۷ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۱۷- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
۱	administrativeState	{smi2AttributeID 31}		-–		x		x	
۲	allomorphs	{smi2AttributeID 50}		x		c1		x	
۳	availabilityStatus	{smi2AttributeID 33}		–		x m		x	
۴	controlStatus	{smi2AttributeID 34}		–		m		x	
۵	launchPadId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx6(6)}		–		m		x	
۶	nameBinding	{smi2AttributeID 63}		–		m		x	
۷	objectClass	{smi2AttributeID 65}		–		m		x	
۸	observedAttributeId	{joint-iso-itu-t ms(9) function(2) part11(11) attribute(7) xx15(15)}		m		m		m	
۹	observedObjectInstance	{joint-iso-itu-t ms(9) function(2) part11(11) attribute(7) xx16(16)}		m		m		m	
۱۰	operationalState	{smi2AttributeID 35}		–		x		x	
۱۱	packages	{smi2AttributeID 66}		–		m		x	
۱۲	schedulerName	{smi2AttributeID 67}		–		m		x	
۱۳	usageState	{smi2AttributeID 39}		–		x		x	

جدول ح-۱۷- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
1	x		x		x		
2	x		x		x		
3	x		x		x		
4	x		x		x		
5	x		x		x		
6	x		x		x		
7	x		x		x		
8	x		x		x		
9	x		x		x		
10	x		x		x		
11	x		x		x		
12	x		x		x		
13	x		x		x		
c1: if not (H-14/1b) then m else							

ح-۴-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۴-۴ عمل‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» از جدول ح-۱۸ بیان کند آیا عمل‌های مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر.

جدول ح-۱۸- MOCS – پشتیبانی از عمل

ردیف	برچسب قالب نوع عمل	مقدار شناسه شیء برای نوع عمل	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	resume	{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx1(1)}		m		
۲	suspend	{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx2(2)}		m		
۳	terminate	{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx3(3)}		m		

جدول ح-۱۹- MOCS – پشتیبانی از عمل

ردیف	زیرردیف	برچسب نام فیلد عمل	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	1.1	SpawnerObjectId		m		
	1.1.1	triggerId		m		
	1.1.1.1	distinguishedName		c:o.1		
	1.1.1.1.1	AttributeType		c:m		
	1.1.1.1.2	AttributeValue		c:m		
	1.1.1.2	nonSpecificForm		c:o.1		
	1.1.1.3	localDistinguishedName		c:o.1		
	1.1.1.3.1	AttributeType		c:m		
	1.1.1.3.2	AttributeValue		c:m		
	1.1.2	CHOICE		m		
	1.1.2.1	threadId		c:o.2		
	1.1.2.1.1	distinguishedName		c:o.3		
	1.1.2.1.1.1	AttributeType		c:m		
	1.1.2.1.1.2	AttributeValue		c:m		
	1.1.2.1.2	nonSpecificForm		c:o.3		
	1.1.2.1.3	localDistinguishedName		c:o.3		
	1.1.2.1.3.1	AttributeType		c:m		
	1.1.2.1.3.2	AttributeValue		c c:m		
	1.1.2.2	launchPadId		c:o.2		
	1.1.2.2.1	distinguishedName		c:m		
	1.1.2.2.1.1	AttributeType		o		
	1.1.2.2.1.2	AttributeValue		c:m		
	1.1.2.2.2	nonSpecificForm		c:o.4		
	1.1.2.2.3	localDistinguishedName		c:o.4		
	1.1.2.2.3.1	AttributeType		c:m		
	1.1.2.2.3.2	AttributeValue		c:m		
2	2.1	SpawnerObjectId		m		
	2.1.1	triggerId		m		
	2.1.1.1	distinguishedName		c:o.5		
	2.1.1.1.1	AttributeType		c:m		
	2.1.1.1.2	AttributeValue		c:m		
	2.1.1.2	nonSpecificForm		c:o.5		
	2.1.1.3	localDistinguishedName		c:o.5		
	2.1.1.3.1	AttributeType		c:m		

		c:m		AttributeValue	2.1.1.3.2	
		m		CHOICE	2.1.2	
		c:o.6		threadId	2.1.2.1	
		c:o.7		distinguishedName	2.1.2.1.1	
		c:m		AttributeType	2.1.2.1.1.1	
		c:m		AttributeValue	2.1.2.1.1.2	
		c:o.7		nonSpecificForm	2.1.2.1.2	
		c:o.7		localDistinguishedName	2.1.2.1.3	
		c:m		AttributeType	2.1.2.1.3.1	
		c:m		AttributeValue	2.1.2.1.3.2	
		c:m		AttributeType	2.1.2.2.1.1	
		c:m		AttributeValue	2.1.2.2.1.2	
		c:o.8		nonSpecificForm	2.1.2.2.2	
		c:o.8		localDistinguishedName	2.1.2.2.3	
		c:m		AttributeType	2.1.2.2.3.1	
		c:m		AttributeValue	2.1.2.2.3.2	
		m		TriggerId	3.1	3
		c:o.9		distinguishedName	3.1.1	
		c:m		AttributeType	3.1.1.1	
		c:m		AttributeValue	3.1.1.2	
		c:o.9		nonSpecificForm	3.1.2	
		c:o.9o		localDistinguishedName	3.1.3	
		c:m		AttributeType	3.1.3.1	
		c:m		AttributeValue	3.1.3.2	

ح-۴-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۴-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۵ بیانیه انطباق با کلاس شیء asynchronousLaunchPad

جدول ح-۲۰- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	asynchronousLaunchPad	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx4(4)}		

اگر پاسخ پرسش کلاس واقعی در پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۲۰ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۲۱ تکمیل کند.

جدول ح-۲۱- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۵-۱ بسته‌ها

تأمین‌کننده‌یپیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۲۲ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۲۲- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-20/1b) then m else –						

ح-۵-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۲۳ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

ح-۵-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

جدول ح-۲۳- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
۱	administrativeState	{smi2AttributeID 31}		-–		x		x	
۲	allomorphs	{smi2AttributeID 50}		x		c1		x	
۳	availabilityStatus	{smi2AttributeID 33}		–		x		x	
۴	controlStatus	{smi2AttributeID 34}		–		m		x	
۵	launchPadId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx6(6)}		–		m		x	
۶	nameBinding	{smi2AttributeID 63}		–		m		x	
۷	objectClass	{smi2AttributeID 65}		–		m		x	
۸	observedAttributeId	{joint-iso-itu-t ms(9) function(2) part11(11) attribute(7) xx15(15)}		m		m		m	
۹	observedObjectInstance	{joint-iso-itu-t ms(9) function(2) part11(11) attribute(7) xx16(16)}		m		m		m	
۱۰	operationalState	{smi2AttributeID 35}		–		x		x	
۱۱	packages	{smi2AttributeID 66}		–		m		x	
۱۲	schedulerName	{smi2AttributeID 67}		–		m		x	
۱۳	usageState	{smi2AttributeID 39}		–		x		x	

جدول ح-۲۳- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
۵	x		x		x		
۶	x		x		x		
۷	x		x		x		
۸	x		x		x		
۹	x		x		x		
۱۰	x		x		x		
۱۱	x		x		x		
۱۲	x		x		x		
۱۳	x		x		x		
c1: if not (H-20/1b) then m else –							

ح-۵-۴ عمل‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» از جدول ح-۲۴ بیان کند آیا عمل‌های مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر.

جدول ح-۲۴- MOCS – پشتیبانی از عمل

ردیف	برچسب قالب نوع عمل	مقدار شناسه شیء برای نوع عمل	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	resume	{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx1(1)}		m		
۲	suspend	{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx2(2)}		m		
۳	terminate	{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx3(3)}		m		

جدول ح-۲۵- MOCS – پشتیبانی از عمل

ردیف	برچسب قالب نوع عمل	مقدار شناسه شیء برای نوع عمل	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	1.1	SpawnerObjectId		m		
	1.1.1	triggerId		m		
	1.1.1.1	distinguishedName		c:o.1		
	1.1.1.1.1	AttributeType		c:m		
	1.1.1.1.2	AttributeValue		c:m		
	1.1.1.2	nonSpecificForm		c:o.1		
	1.1.1.3	localDistinguishedName		c:o.1		
	1.1.1.3.1	AttributeType		c:m		
	1.1.1.3.2	AttributeValue		c:m		
	1.1.2	CHOICE		M		
	1.1.2.1	threadId		c:o.2		
	1.1.2.1.1	distinguishedName		c:o.3		
	1.1.2.1.1.1	AttributeType		c:m		
	1.1.2.1.1.2	AttributeValue		c:m		
	1.1.2.1.2	nonSpecificForm		c:o.3		
	1.1.2.1.3	localDistinguishedName		c:o.3		
	1.1.2.1.3.1	AttributeType		c:m		
	1.1.2.1.3.2	AttributeValue		c c:m		
	1.1.2.2	launchPadId		c:o.2		
	1.1.2.2.1	distinguishedName		c:m		
	1.1.2.2.1.1	AttributeType		o		
	1.1.2.2.1.2	AttributeValue		c:m		
	1.1.2.2.2	nonSpecificForm		c:o.4		
	1.1.2.2.3	localDistinguishedName		c:o.4		
	1.1.2.2.3.1	AttributeType		c:m		
	1.1.2.2.3.2	AttributeValue		c:m		
2	2.1	SpawnerObjectId		m		
	2.1.1	triggerId		m		
	2.1.1.1	distinguishedName		c:o.5		
	2.1.1.1.1	AttributeType		c:m		
	2.1.1.1.2	AttributeValue		c:m		
	2.1.1.2	nonSpecificForm		c:o.5		
	2.1.1.3	localDistinguishedName		c:o.5		

		c:m		AttributeType	2.1.1.3.1	
		c:m		AttributeValue	2.1.1.3.2	
		m		CHOICE	2.1.2	
		c:o.6		threadId	2.1.2.1	
		c:o.7		distinguishedName	2.1.2.1.1	
		c:m		AttributeType	2.1.2.1.1.1	
		c:m		AttributeValue	2.1.2.1.1.2	
		c:o.7		nonSpecificForm	2.1.2.1.2	
		c:o.7		localDistinguishedName	2.1.2.1.3	
		c:m		AttributeType	2.1.2.1.3.1	
		c:m		AttributeValue	2.1.2.1.3.2	
		c:m		AttributeType	2.1.2.2.1.1	
		c:m		AttributeValue	2.1.2.2.1.2	
		c:o.8		nonSpecificForm	2.1.2.2.2	
		c:o.8		localDistinguishedName	2.1.2.2.3	
		c:m		AttributeType	2.1.2.2.3.1	
		c:m		AttributeValue	2.1.2.2.3.2	
		m		TriggerId	3.1	3
		c:o.9		distinguishedName	3.1.1	
		c:m		AttributeType	3.1.1.1	
		c:m		AttributeValue	3.1.1.2	
		c:o.9		nonSpecificForm	3.1.2	
		c:o.9o		localDistinguishedName	3.1.3	
		c:m		AttributeType	3.1.3.1	
		c:m		AttributeValue	3.1.3.2	

ح-۵-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۵-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۶ بیانیه انطباق با کلاس شیء synchronousLaunchPad

جدول ح-۲۶- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	synchronousLaunchPad	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx5(5)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۲۶ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۲۷ تکمیل کند.

جدول ح-۲۷- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۶-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۲۸ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۲۸- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-26/1b) then m else –						

ح-۶-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۲۹ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی نشان دهد.

جدول ح-۲۹- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	administrativeState	{smi2AttributeID 31}		-–		x		x	
2	allomorphs	{smi2AttributeID 50}		x		c1		x	
3	availabilityStatus	{smi2AttributeID 33}		–		x		x	
4	controlStatus	{smi2AttributeID 34}		–		m		x	
5	launchPadId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx6(6)}		–		m		x	
6	nameBinding	{smi2AttributeID 63}		–		m		x	
7	objectClass	{smi2AttributeID 65}		–		m		x	
8	observedAttributeId	{joint-iso-itu-t ms(9) function(2) part11(11) attribute(7) xx15(15)}		m		m		m	
9	observedObjectInstance	{joint-iso-itu-t ms(9) function(2) part11(11) attribute(7) xx16(16)}		m		m		m	
10	operationalState	{smi2AttributeID 35}		–		x		x	
11	packages	{smi2AttributeID 66}		–		m		x	
12	schedulerName	{smi2AttributeID 67}		–		m		x	
13	usageState	{smi2AttributeID 39}		–		x		x	

جدول ح-۲۹- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
۵	x		x		x		
۶	x		x		x		
۷	x		x		x		
۸	x		x		x		
۹	x		x		x		
۱۰	x		x		x		
۱۱	x		x		x		
۱۲	x		x		x		
۱۳	x		x		x		
c1: if not (H-26/1b) then m else –							

ح-۶-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۶-۴ عمل‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» از جدول ح-۳۰ بیان کند آیا عمل‌های مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر.

جدول ح-۳۰- MOCS – پشتیبانی از عمل

ردیف	برچسب قالب نوع عمل	مقدار شناسه شیء برای نوع عمل	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	resume	{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx1(1)}		m		
۲	suspend	{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx2(2)}		m		
۳	terminate	{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx3(3)}		m		

جدول ح-۳۱- MOCS – پشتیبانی از عمل

ردیف	برچسب قالب نوع عمل	مقدار شناسه شیء برای نوع عمل	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	1.1	SpawnerObjectId		m		
	1.1.1	triggerId		m		
	1.1.1.1	distinguishedName		c:o.1		
	1.1.1.1.1	AttributeType		c:m		
	1.1.1.1.2	AttributeValue		c:m		
	1.1.1.2	nonSpecificForm		c:o.1		
	1.1.1.3	localDistinguishedName		c:o.1		
	1.1.1.3.1	AttributeType		c:m		
	1.1.1.3.2	AttributeValue		c:m		
	1.1.2	CHOICE		M		
	1.1.2.1	threadId		c:o.2		
	1.1.2.1.1	distinguishedName		c:o.3		
	1.1.2.1.1.1	AttributeType		c:m		
	1.1.2.1.1.2	AttributeValue		c:m		
	1.1.2.1.2	nonSpecificForm		c:o.3		
	1.1.2.1.3	localDistinguishedName		c:o.3		
	1.1.2.1.3.1	AttributeType		c:m		
	1.1.2.1.3.2	AttributeValue		c c:m		
	1.1.2.2	launchPadId		c:o.2		
	1.1.2.2.1	distinguishedName		c:m		
	1.1.2.2.1.1	AttributeType		o		
	1.1.2.2.1.2	AttributeValue		c:m		
	1.1.2.2.2	nonSpecificForm		c:o.4		
	1.1.2.2.3	localDistinguishedName		c:o.4		
	1.1.2.2.3.1	AttributeType		c:m		
	1.1.2.2.3.2	AttributeValue		c:m		
2	2.1	SpawnerObjectId		m		
	2.1.1	triggerId		m		
	2.1.1.1	distinguishedName		c:o.5		
	2.1.1.1.1	AttributeType		c:m		
	2.1.1.1.2	AttributeValue		c:m		
	2.1.1.2	nonSpecificForm		c:o.5		
	2.1.1.3	localDistinguishedName		c:o.5		

		c:m		AttributeType	2.1.1.3.1	
		c:m		AttributeValue	2.1.1.3.2	
		m		CHOICE	2.1.2	
		c:o.6		threadId	2.1.2.1	
		c:o.7		distinguishedName	2.1.2.1.1	
		c:m		AttributeType	2.1.2.1.1.1	
		c:m		AttributeValue	2.1.2.1.1.2	
		c:o.7		nonSpecificForm	2.1.2.1.2	
		c:o.7		localDistinguishedName	2.1.2.1.3	
		c:m		AttributeType	2.1.2.1.3.1	
		c:m		AttributeValue	2.1.2.1.3.2	
		c:m		AttributeType	2.1.2.2.1.1	
		c:m		AttributeValue	2.1.2.2.1.2	
		c:o.8		nonSpecificForm	2.1.2.2.2	
		c:o.8		localDistinguishedName	2.1.2.2.3	
		c:m		AttributeType	2.1.2.2.3.1	
		c:m		AttributeValue	2.1.2.2.3.2	
		m		TriggerId	3.1	3
		c:o.9		distinguishedName	3.1.1	
		c:m		AttributeType	3.1.1.1	
		c:m		AttributeValue	3.1.1.2	
		c:o.9		nonSpecificForm	3.1.2	
		c:o.9o		localDistinguishedName	3.1.3	
		c:m		AttributeType	3.1.3.1	
		c:m		AttributeValue	3.1.3.2	

ح-۶-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۶-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۷ بیانیه انطباق با کلاس شیء launchScript

جدول ح-۳۲- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	launchScript	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx7(7)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۳۲ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۳۳ تکمیل کند.

جدول ح-۳۳- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۷-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۳۴ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۳۴- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-32/1b) then m else – c2: if H-34/1 then m else –						

ح-۷-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۳۵ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۳۵- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	administrativeState	{smi2AttributeID 31}		-–		x		x	
2	allomorphs	{smi2AttributeID 50}		x		c1		x	
3	executionResultType	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx3(3)}		–		m		x	
4	nameBinding	{smi2AttributeID 63}		–		m		x	
5	launchPadId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx6(6)}		–		m		x	
6	nameBinding	{smi2AttributeID 63}		–		m		x	
7	scriptId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx5(5)}		–		m		x	

جدول ح-۳۵- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
۵	x		x		x		
۶	x		x		x		
۷	x		x		x		

c1: if not (H-32/1b) then m else –
c2: if H.34/2 then m else –

ح-۷-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۷-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۷-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۷-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۸ بیانیه انطباق با کلاس شیء scriptReferencer

جدول ح-۳۶- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	scriptReferencer	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx8(8)}		

اگر پاسخ پرسش کلاس واقعی در پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۳۶ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۳۷ تکمیل کند.

جدول ح-۳۷- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۸-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۳۸ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۳۸- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-36/1b) then m else –						

ح-۸-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۳۹ بیان کند آیا صفات مشخص شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۳۹- MOCS – پشتیبانی از صفت

			تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	allomorphs	{smi2Attribute ID 50}		x		c1		x	
2	nameBinding	{smi2Attribute ID 63}		–		m		x	
3	objectClass	{smi2Attribute ID 65}		–		m		x	
4	packages	{smi2Attribute ID 66}		–		m		x	

جدول ح-۳۹- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
c1: if not (H-36/1b) then m else –							

ح-۸-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۸-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۸-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۸-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۹ بیانیه انطباق با کلاس شیء thread

جدول ح-۴۰- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	thread	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx9(9)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۴۰ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۴۱ تکمیل کند.

جدول ح-۴۱- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۹-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۴۲ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۴۲- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-40/1b) then m else –						

ح-۹-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۴۳ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی

می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۴۳- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	allomorphs	{smi2AttributeID 50}		x		c1		x	
2	executingParameters	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx9(9)}		–		m		x	
3	nameBinding	{smi2AttributeID 63}		–		m		x	
4	objectClass	{smi2AttributeID 65}		–		m		x	
5	operationalState	{smi2AttributeID 35}		–		m		x	
6	packages	{smi2AttributeID 66}		–		m		x	
7	scriptId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx5(5)}		–		m		x	
8	threadId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx10(10)}		–		m		x	

جدول ح-۴۳- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
۵	x		x		x		
۶	x		x		x		

		x		x		x	۷
		x		x		x	۸
c1: if not (H-40/1b) then m else –							

ح-۹-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۹-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۹-۵ اعلان‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» از جدول ح-۴۴ بیان کند که آیا اعلان‌های مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی را در قالب مدهای تأییدشده و تأییدنشده نشان دهد.

جدول ح-۴۴- MOCS – پشتیبانی از اعلان

	پشتیبانی						
اطلاعات افزوده	**تأییدنشده**	**تأییدشده**	**وضعیت**	**قیود و مقادیر**	**مقدار شناسه شیء برای نوع اعلان**	**برچسب قالب نوع اعلان**	**ردیف**
			m		{smi2Notification 10}	processingErr orAlarm	۱

جدول ح-۴۴- (نهایی)

اطلاعات افزوده	**پشتیبانی**	**وضعیت**	**قیود و مقادیر**	**مقدار شناسه شیء نوع صفت مرتبط با فیلد**	**برچسب نام فیلد اعلان**	**زیرردیف**	**ردیف**
		o		{smi2AttributeID 6}	additionalInformation	1.1	1
		c:m		–	identifier	1.1.1	
		c:m		–	significance	1.1.2	
		c:m		–	information	1.1.3	
		o		{smi2AttributeID 7}	additionalText	1.2	
		o		{smi2AttributeID 11}	backedUpStatus	1.3	
		o		{smi2AttributeID 40}	backUpObject	1.4	
		c:o.1		–	objectName	1.4.1	
		c:o.2		–	distinguishedName	1.4.1.1	
		c:m		–	AttributeType	1.4.1.1.1	
		c:m		–	AttributeValue	1.4.1.1.2	

		c:o.2		–	nonSpecificForm	1.4.1.2	
		c:o.2		–	localDistinguishedName	1.4.1.3	
		c:m		–	AttributeType	1.4.1.3.1	
		c:m		–	AttributeValue	1.4.1.3.2	
		c:o.1		–	noObject	1.4.2	
		o		{smi2AttributeID 12}	correlatedNotifications	1.5	
		c:m		–	correlatedNotifications	1.5.1	
		c:o		–	sourceObjectInst	1.5.2	
		c:o.3		–	distinguishedName	1.5.2.1	
		c:m		–	AttributeType	1.5.2.1.1	
		c:m		–	AttributeValue	1.5.2.1.2	
		c:o.3		–	nonSpecificForm	1.5.2.2	
		c:o.3		–	localDistinguishedName	1.5.2.3	
		c:m		–	AttributeType	1.5.2.3.1	
		c:m		–	AttributeValue	1.5.2.3.2	
		o		{smi2AttributeID 15}	monitoredAttributes	1.6	
		c:m		–	attributeId	1.6.1	
		c:o.4		–	globalForm	1.6.1.1	
		c:o.4		–	localForm	1.6.1.2	
		c:m		–	attributeValue	1.6.2	
		o		{smi2AttributeID 16}	notificationIdentifier	1.7	
		m		{smi2AttributeID 17}	perceivedSeverity	1.8	
		m		{smi2AttributeID 18}	probableCause	1.9	
		c:o.5		–	globalValue	1.9.1	
		c:o.5		–	localValue	1.9.2	
		o		{smi2AttributeID 19}	proposedRepairActions	1.10	
		c:o.6		–	OBJECT IDENTIFIER	1.10.1	
		c:o.6		–	INTEGER	1.10.2	
		o		{smi2AttributeID 27}	specificProblems	1.11	
		c:o.7		–	OBJECT IDENTIFIER	1.11.1	
		c:o.7		–	INTEGER	1.1.2	
		o		{smi2AttributeID 28}	stateChangeDefinition	1.12	
		c:m		–	attributeID	1.12.1	
		c:o.8		–	globalForm	1.12.1.1	
		c:o.8		–	localForm	1.12.1.2	
		c:o		–	oldAttributeValue	1.12.2	
		c:m		–	newAttributeValue	1.12.3	
		o		{smi2AttributeID 29}	thresholdInfo	1.13	
		c:m		–	triggeredThreshold	1.13.1	
		c:o.9		–	globalForm	1.13.1.1	
		c:o.9		–	localForm	1.13.1.2	
		c:m		–	observedValue	1.13.2	

		c:o.10		–	integer	1.13.2.1	
		c:o.10		–	real	1.13.2.2	
		c:o		–	thresholdLevel	1.13.3	
		c:o.11		–	up	1.13.3.1	
		c:m		–	high	1.13.3.1.1	
		c:o.12		–	integer	1.13.3.1.1.1	
		c:o.12		–	real	1.13.3.1.1.2	
		c:o		–	low	1.13.3.1.2	
		c:o.13		–	integer	1.13.3.1.2.1	
		c:o.13		–	real	1.13.3.1.2.2	
		c:o.11		–	down	1.13.3.2	
		c:m		–	high	1.13.3.2.1	
		c:o.14		–	integer	1.13.3.2.1.1	
		c:o.14		–	real	1.13.3.2.1.2	
		c:m		–	low	1.13.3.2.2	
		c:o.15		–	integer	1.13.3.2.2.1	
		c:o.15		–	real	1.13.3.2.2.2	
		c:o		–	armTime	1.13.4	
		o		{smi2AttributeID 30}	trendIndication	1.14	

ح-۹-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۱۰ بیانیه انطباق با کلاس شیء suspendableThread

جدول ح-۴۵- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	suspendableThread	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx10(10)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۴۵ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۴۶ تکمیل کند.

جدول ح-۴۶- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۱۰-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۴۷ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۴۷- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-45/1b) then m else						

ح-۱۰-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۴۸ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۴۸- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	allomorphs	{smi2Attribute ID 50}		x		c1		x	
2	controlStatus	{smi2Attribute ID 34}		–		m		x	
3	executingParameters	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx9(9)}		–		m		x	
4	nameBinding	{smi2Attribute ID 63}		–		m		x	
5	objectClass	{smi2Attribute ID 65}		–		m		x	
6	operationalState	{smi2Attribute ID 35}		–		m		x	
7	packages	{smi2Attribute ID 66}		–		m		x	
8	scriptId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7)		–		m		x	

							xx5(5)}		
	x		m		–		{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx10(10)}	threadId	9

جدول ح-۴۸- (نهایی)

	تنظیم به پیش‌فرض		حذف کردن		اضافه کردن		
اطلاعات افزوده	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	ردیف
		x		x		x	۱
		x		x		x	۲
		x		x		x	۳
		x		x		x	۴
		x		x		x	۵
		x		x		x	۶
		x		x		x	۷
		x		x		x	۸
		x		x		x	۹
c1: if not (H-45/1b) then m else –							

ح-۱۰-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۱۰-۴ عمل‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» از جدول ح-۴۹ بیان کند آیا عمل‌های مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر.

جدول ح-۴۹- MOCS – پشتیبانی از عمل

اطلاعات افزوده	پشتیبانی	وضعیت	قیود و مقادیر	مقدار شناسه شیء برای نوع عمل	برچسب قالب نوع عمل	ردیف
		m		{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx1(1)}	resume	۱
		m		{joint-iso-itu-t ms(9) function(2) part21(21) action(9) xx2(2)}	suspend	۲

ح-۱۰-۵ اعلان‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» از جدول ح-۵۱ بیان کند که آیا اعلان‌های مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی را در قالب مدهای تأییدشده و تأییدنشده نشان دهد.

جدول ح-۵۰- MOCS – پشتیبانی از عمل

ردیف	برچسب قالب نوع عمل	مقدار شناسه شیء برای نوع عمل	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	1.1	SpawnerObjectId		m		
	1.1.1	triggerId		m		
	1.1.1.1	distinguishedName		c:o.1		
	1.1.1.1.1	AttributeType		c:m		
	1.1.1.1.2	AttributeValue		c:m		
	1.1.1.2	nonSpecificForm		c:o.1		
	1.1.1.3	localDistinguishedName		c:o.1		
	1.1.1.3.1	AttributeType		c:m		
	1.1.1.3.2	AttributeValue		c:m		
	1.1.2	CHOICE		M		
	1.1.2.1	threadId		c:o.2		
	1.1.2.1.1	distinguishedName		c:o.3		
	1.1.2.1.1.1	AttributeType		c:m		
	1.1.2.1.1.2	AttributeValue		c:m		
	1.1.2.1.2	nonSpecificForm		c:o.3		
	1.1.2.1.3	localDistinguishedName		c:o.3		
	1.1.2.1.3.1	AttributeType		c:m		
	1.1.2.1.3.2	AttributeValue		c c:m		
	1.1.2.2	launchPadId		c:o.2		
	1.1.2.2.1	distinguishedName		c:m		
	1.1.2.2.1.1	AttributeType		o		
	1.1.2.2.1.2	AttributeValue		c:m		
	1.1.2.2.2	nonSpecificForm		c:o.4		
	1.1.2.2.3	localDistinguishedName		c:o.4		
	1.1.2.2.3.1	AttributeType		c:m		
	1.1.2.2.3.2	AttributeValue		c:m		
2	2.1	SpawnerObjectId		m		

				m	triggerId	2.1.1
				c:o.5	distinguishedName	2.1.1.1
				c:m	AttributeType	2.1.1.1.1
				c:m	AttributeValue	2.1.1.1.2
				c:o.5	nonSpecificForm	2.1.1.2
				c:o.5	localDistinguishedName	2.1.1.3
				c:m	AttributeType	2.1.1.3.1
				c:m	AttributeValue	2.1.1.3.2
				m	CHOICE	2.1.2
				c:o.6	threadId	2.1.2.1
				c:o.7	distinguishedName	2.1.2.1.1
				c:m	AttributeType	2.1.2.1.1.1
				c:m	AttributeValue	2.1.2.1.1.2
				c:o.7	nonSpecificForm	2.1.2.1.2
				c:o.7	localDistinguishedName	2.1.2.1.3
				c:m	AttributeType	2.1.2.1.3.1
				c:m	AttributeValue	2.1.2.1.3.2
				c:m	AttributeType	2.1.2.2.1.1
				c:m	AttributeValue	2.1.2.2.1.2
				c:o.8	nonSpecificForm	2.1.2.2.2
				c:o.8	localDistinguishedName	2.1.2.2.3
				c:m	AttributeType	2.1.2.2.3.1
				c:m	AttributeValue	2.1.2.2.3.2

جدول ح-۵۱- MOCS – پشتیبانی از اعلان

	پشتیبانی						
اطلاعات افزوده	تأییدنشده	تأییدشده	وضعیت	قیود و مقادیر	مقدار شناسه شیء برای نوع اعلان	برچسب قالب نوع اعلان	ردیف
			m		{smi2Notification 10}	processingErrorAlarm	۱

جدول ح-۵۱- (نهایی)

اطلاعات افزوده	پشتیبانی	وضعیت	قیود و مقادیر	مقدار شناسه شیء نوع صفت مرتبط با فیلد	برچسب نام فیلد اعلان	زیرردیف	ردیف
		o		{smi2AttributeID 6}	additionalInformation	1.1	1
		c:m		–	identifier	1.1.1	
		c:m		–	significance	1.1.2	
		c:m		–	information	1.1.3	

		o		{smi2AttributeID 7}	additionalText	1.2	
		o		{smi2AttributeID 11}	backedUpStatus	1.3	
		o		{smi2AttributeID 40}	backUpObject	1.4	
		c:o.1		–	objectName	1.4.1	
		c:o.2		–	distinguishedName	1.4.1.1	
		c:m		–	AttributeType	1.4.1.1.1	
		c:m		–	AttributeValue	1.4.1.1.2	
		c:o.2		–	nonSpecificForm	1.4.1.2	
		c:o.2		–	localDistinguishedName	1.4.1.3	
		c:m		–	AttributeType	1.4.1.3.1	
		c:m		–	AttributeValue	1.4.1.3.2	
		c:o.1		–	noObject	1.4.2	
		o		{smi2AttributeID 12}	correlatedNotifications	1.5	
		c:m		–	correlatedNotifications	1.5.1	
		c:o		–	sourceObjectInst	1.5.2	
		c:o.3		–	distinguishedName	1.5.2.1	
		c:m		–	AttributeType	1.5.2.1.1	
		c:m		–	AttributeValue	1.5.2.1.2	
		c:o.3		–	nonSpecificForm	1.5.2.2	
		c:o.3		–	localDistinguishedName	1.5.2.3	
		c:m		–	AttributeType	1.5.2.3.1	
		c:m		–	AttributeValue	1.5.2.3.2	
		o		{smi2AttributeID 15}	monitoredAttributes	1.6	
		c:m		–	attributeId	1.6.1	
		c:o.4		–	globalForm	1.6.1.1	
		c:o.4		–	localForm	1.6.1.2	
		c:m		–	attributeValue	1.6.2	
		o		{smi2AttributeID 16}	notificationIdentifier	1.7	
		m		{smi2AttributeID 17}	perceivedSeverity	1.8	
		m		{smi2AttributeID 18}	probableCause	1.9	
		c:o.5		–	globalValue	1.9.1	
		c:o.5		–	localValue	1.9.2	
		o		{smi2AttributeID 19}	proposedRepairActions	1.10	
		c:o.6		–	OBJECT IDENTIFIER	1.10.1	
		c:o.6		–	INTEGER	1.10.2	
		o		{smi2AttributeID 27}	specificProblems	1.11	
		c:o.7		–	OBJECT IDENTIFIER	1.11.1	
		c:o.7		–	INTEGER	1.11.2	
		o		{smi2AttributeID	stateChangeDefinition	1.12	

				28}			
		c:m		–	attributeID	1.12.1	
		c:o.8		–	globalForm	1.12.1.1	
		c:o.8		–	localForm	1.12.1.2	
		c:o		–	oldAttributeValue	1.12.2	
		c:m		–	newAttributeValue	1.12.3	
		o		{smi2AttributeID 29}	thresholdInfo	1.13	
		c:m		–	triggeredThreshold	1.13.1	
		c:o.9		–	globalForm	1.13.1.1	
		c:o9		–	localForm	1.13.1.2	
		c:m		–	observedValue	1.13.2	
		c:o.10		–	integer	1.13.2.1	
		c:o.10		–	real	1.13.2.2	
		c:o		–	thresholdLevel	1.13.3	
		c:o.11		–	up	1.13.3.1	
		c:m		–	high	1.13.3.1.1	
		c:o.12		–	integer	1.13.3.1.1.1	
		c:o.12		–	real	1.13.3.1.1.2	
		c:o		–	low	1.13.3.1.2	
		c:o.13		–	integer	1.13.3.1.2.1	
		c:o.13		–	real	1.13.3.1.2.2	
		c:o.11		–	down	1.13.3.2	
		c:m		–	high	1.13.3.2.1	
		c:o.14		–	integer	1.13.3.2.1.1	
		c:o.14		–	real	1.13.3.2.1.2	
		c:m		–	low	1.13.3.2.2	
		c:o.15		–	integer	1.13.3.2.2.1	
		c:o.15		–	real	1.13.3.2.2.2	
		c:o		–	armTime	1.13.4	
		o		{smi2AttributeID 30}	trendIndication	1.14	

ح-۱۰-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۱۱ بیانیه انطباق با کلاس شیء eventDiscriminationCounter

جدول ح-۵۲- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	eventDiscriminationCounter	{joint-iso-itu-t ms(9) ms(9) function(2) part21(21) managedObjectClass(3) xx11(11)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۵۲ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۵۳ تکمیل کند.

جدول ح-۵۳- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۱۱-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۵۴ بیان کند که آیا بسته‌های شرطی مشخص شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۵۴- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	allomorphicPackage	{smi2Package 17}		c1		
2	availabilityStatusPackage	{smi2Package 22}		c2		
3	counterAlarmPackage	{joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx15(15)}		c3		
4	dailyScheduling	{smi2Package 25}		o		
5	duration	{smi2Package 26}		c4		
6	externalScheduler	{smi2Package 27		o		
7	packagesPackage	{smi2Package 16}		c5		
8	weeklyScheduling	{smi2Package 29}		o		

c1: if not (H-52/1b) then m else –
c2: if "any of the scheduling packages, (duration, weekly scheduling, external) are present" then m else –
c3: if "a counter is of finite size and a notification is triggered by a capacity alarm threshold" then m else –
c4: if "the discriminator function is scheduled to start at a specified time and stop at either a specified time or function continuously" then m else –
c5: if H-54/1 or H-54/2 or H-54/3 or H-54/4 or H-54/5 or H-54/6 or H-54/8 then m else –

ح-۱۱-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۵۵ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

ح-۱۱-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۱۱-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۱۱-۵ اعلان‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» از جدول ح-۵۶ بیان کند که آیا اعلان‌های مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی را در قالب مدهای تأییدشده و تأییدنشده نشان دهد.

جدول ح-۵۵- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	administrativeState	{smi2AttributeID 31}		m		m		m	
2	allomorphs	{smi2AttributeID 50}		x		c1		x	
3	availabilityStatus	{smi2AttributeID 33}		–		c2		x	
4	capacityAlarmThreshold	{smi2AttributeID 52}		c3		c3		c3	
5	counter	{smi2AttributeID 88}		–		m		x	
6	discriminatorConstruct	{smi2AttributeID 56}		m		m		m	
7	discriminatorId	{smi2AttributeID 1}		–		m		x	
8	intervalsOfDay	{smi2AttributeID 57}		o		o			
9	maxCounterSize	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx12(12)}		–		m		x	
10	nameBinding	{smi2AttributeID 63}		–		m		x	
11	objectClass	{smi2AttributeID 65}		–		m		x	

	x		m		–		{smi2Attribute ID 35}	operational State	12
	x		c4		–		{smi2Attribute ID 66}	packages	13
	x		o		–		{smi2Attribute ID 67}	schedulerName	14
	c5		c5		c5		{smi2Attribute ID 68}	startTime	15
	c5		c5		c5		{smi2Attribute ID 69}	stopTime	16
	o		o		o		{smi2Attribute ID 71}	weekMask	17

جدول ح-۵۵- (نهایی)

	تنظیم به پیش‌فرض		حذف کردن		اضافه کردن		
اطلاعات افزوده	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	ردیف
		x		x		X	۱
		x		x		X	۲
		x		x		x	۳
		x		c3		c3	۴
		x		x		x	۵
		m		x		x	۶
		x		x		x	۷
		o		o		o	۸
		x		x		x	۹
		x		x		x	۱۰
		x		x		x	۱۱
		x		x		x	۱۲
		x		x		x	۱۳
		x		x		x	۱۴
		x		x		x	۱۵
		c5		x		x	۱۶
		o		o		o	۱۷

c1: if not (H-52/1b) then m else –
c2: if H-54/2 then m else –
c3: if H-54/3 then m else –
c4: if H-54/7 then m else –
c5: if H-54/5 then m else –

جدول ح-۵۶- MOCS – پشتیبانی از اعلان

ردیف	برچسب قالب نوع اعلان	مقدار شناسه شیء برای نوع اعلان	قیود و مقادیر	وضعیت	پشتیبانی		اطلاعات افزوده
					تأییدشده	تأییدنشده	
1	attributeValueChange	{smi2Notification 1}		m			
2	objectCreation	{smi2Notification 6}		m			
3	objectDeletion	{smi2Notification 7}		m			
4	processingErrorAlarm	{smi2Notification 10}		m			
5	stateChange	{smi2Notification 14}		m			

جدول ح-۵۶- (ادامه)

ردیف	زیرردیف	برچسب نام فیلد اعلان	مقدار شناسه شیء نوع صفت مرتبط با فیلد	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	1.1	additionalInformation	{smi2AttributeID 6}		o		
	1.1.1	identifier	–		c:m		
	1.1.2	significance	–		c:m		
	1.1.3	information	–		c:m		
	1.2	additionalText	{smi2AttributeID 7}		o		
	1.3	attributeIdentifierList	{smi2AttributeID 8}		o		
	1.3.1	globalForm	–		c:o.1		
	1.3.2	localForm	–		c:o.1		
	1.4	attributeValueChangeDefinition	{smi2AttributeID 10}		m		
	1.4.1	attributeID	–		m		
	1.4.1.1	globalForm	–		c:o.2		
	1.4.1.2	localForm	–		c:o.2		
	1.4.2	oldAttributeValue	–		o		
	1.4.3	newAttributeValue	–		m		
	1.5	correlatedNotifications	{smi2AttributeID 12}		o		
	1.5.1	correlatedNotifications	–		c:m		
	1.5.2	sourceObjectInst	–		c:o		
	1.5.2.1	distinguishedName	–		c:o.3		
	1.5.2.1.1	AttributeType	–		c:m		
	1.5.2.1.2	AttributeValue	–		c:m		

		c:o.3		–	nonSpecificForm	1.5.2.2	
		c:o.3		–	localDistinguishedName	1.5.2.3	
		c:m		–	AttributeType	1.5.2.3.1	
		c:m		–	AttributeValue	1.5.2.3.2	
		o		{smi2AttributeID 16}	notificationIdentifier	1.6	
		o		{smi2AttributeID 26}	sourceIndicator	1.7	
		o		{smi2AttributeID 6}	additionalInformation	2.1	2
		c:m		–	identifier	2.1.1	
		c:m		–	significance	2.1.2	
		c:m		–	information	2.1.3	
		o		{smi2AttributeID 7}	additionalText	2.2	
		o		{smi2AttributeID 9}	attributeList	2.3	
		c:m		–	attributeId	2.3.1	
		c:o.4		–	globalForm	2.3.1.1	
		c:o.4		–	localForm	2.3.1.2	
		c:m		–	attributeValue	2.3.2	
		o		{smi2AttributeID 12}	correlatedNotifications	2.4	
		c:m		–	correlatedNotifications	2.4.1	
		c:o		–	sourceObjectInst	2.4.2	
		c:o.5		–	distinguishedName	2.4.2.1	
		c:m		–	AttributeType	2.4.2.1.1	
		c:m		–	AttributeValue	2.4.2.1.2	
		c:o.5		–	nonSpecificForm	2.4.2.2	
		c:o.5		–	localDistinguishedName	2.4.2.3	
		c:m		–	AttributeType	2.4.2.3.1	
		c:m		–	AttributeValue	2.4.2.3.2	
		o		{smi2AttributeID 16}	notificationIdentifier	2.5	
		o		{smi2AttributeID 26}	sourceIndicator	2.6	
		o		{smi2AttributeID 6}	additionalInformation	3.1	3
		c:m		–	identifier	3.1.1	
		c:m		–	significance	3.1.2	
		c:m		–	information	3.1.3	
		o		{smi2AttributeID 7}	additionalText	3.2	
		o		{smi2AttributeID 9}	attributeList	3.3	
		c:m		–	attributeId	3.3.1	
		c:o.6		–	globalForm	3.3.1.1	
		c:o.6		–	localForm	3.3.1.2	
		c:m		–	attributeValue	3.3.2	
		o		{smi2AttributeI	correlatedNotifications	3.4	

				D 12}			
		c:m		–	correlatedNotifications	3.4.1	
		c:o		–	sourceObjectInst	3.4.2	
		c:o.7		–	distinguishedName	3.4.2.1	
		c:m		–	AttributeType	3.4.2.1.1	
		c:m		–	AttributeValue	3.4.2.1.2	
		c:o.7		–	nonSpecificForm	3.4.2.2	
		c:o.7		–	localDistinguishedNa me	3.4.2.3	
		c:m		–	AttributeType	3.4.2.3.1	
		c:m		–	AttributeValue	3.4.2.3.2	
		o		{smi2AttributeI D 16}	notificationIdentifier	3.5	
		o		{smi2AttributeI D 26}	sourceIndicator	3.6	
		o		{smi2AttributeI D 6}	additionalInformation	4.1	
		c:m		–	identifier	4.1.1	
		c:m		–	significance	4.1.2	
		c:m		–	information	4.1.3	
		o		{smi2AttributeI D 7}	additionalText	4.2	
		o		{smi2AttributeI D 11}	backedUpStatus	4.3	
		o		{smi2AttributeI D 40}	backUpObject	4.4	
		c:o.8		–	objectName	4.4.1	
		c:o.9		–	distinguishedName	4.4.1.1	
		c:m		–	AttributeType	4.4.1.1.1	
		c:m		–	AttributeValue	4.4.1.1.2	
		c:o.9		–	nonSpecificForm	4.4.1.2	
		c:o.9		–	localDistinguishedNa me	4.4.1.3	
		c:m		–	AttributeType	4.4.1.3.1	4
		c:m		–	AttributeValue	4.4.1.3.2	
		c:o.8		–	noObject	4.4.2	
		o		{smi2AttributeI D 12}	correlatedNotifications	4.5	
		c:m		–	correlatedNotifications	4.5.1	
		c:o		–	sourceObjectInst	4.5.2	
		c:o.10		–	distinguishedName	4.5.2.1	
		c:m		–	AttributeType	4.5.2.1.1	
		c:m		–	AttributeValue	4.5.2.1.2	
		c:o.10		–	nonSpecificForm	4.5.2.2	
		c:o.10		–	localDistinguishedNa me	4.5.2.3	
		c:m		–	AttributeType	4.5.2.3.1	
		c:m		–	AttributeValue	4.5.2.3.2	
		o		{smi2AttributeI D 15}	monitoredAttributes	4.6	
		c:m		–	attributeId	4.6.1	

		c:o.11		–	globalForm	4.6.1.1	
		c:o.11		–	localForm	4.6.1.2	
		c:m		–	attributeValue	4.6.2	
		o		{smi2AttributeID 16}	notificationIdentifier	4.7	
		m		{smi2AttributeID 17}	perceivedSeverity	4.8	
		m		{smi2AttributeID 18}	probableCause	4.9	
		c:o.12		–	globalValue	4.9.1	
		c:o.12		–	localValue	4.9.2	
		o		{smi2AttributeID 19}	proposedRepairActions	4.10	
		c:o.13		–	OBJECT IDENTIFIER	4.10.1	
		c:o.13		–	INTEGER	4.10.2	
		o		{smi2AttributeID 27}	specificProblems	4.11	
		c:o.14		–	OBJECT IDENTIFIER	4.11.1	
		c:o.14		–	INTEGER	4.11.2	
		o		{smi2AttributeID 28	stateChangeDefinition	4.12	
		c:m		–	attributeID	4.12.1	
		c:o.15		–	globalForm	4.12.1.1	
		c:o.15		–	localForm	4.12.1.2	
		c:o		–	oldAttributeValue	4.12.2	
		c:m		–	newAttributeValue	4.12.3	
		o		{smi2AttributeID29}	thresholdInfo	4.13	
		c:m		–	triggeredThreshold	4.13.1	
		c:o.16		–	globalForm	4.13.1.1	
		c:o.16		–	localForm	4.13.1.2	
		c:m		–	observedValue	4.13.2	
		c:o.17		–	integer	4.13.2.1	
		c:o.17		–	real	4.13.2.2	
		c:o		–	thresholdLevel	4.13.3	
		c:o.18		–	up	4.13.3.1	
		c:m		–	high	4.13.3.1.1	
		c:o.19		–	integer	4.13.3.1.1.1	
		c:o.19		–	real	4.13.3.1.1.2	
		c:o		–	low	4.13.3.1.2	
		c:o.20		–	integer	4.13.3.1.2.1	
		c:o.2		–	real	4.13.3.1.2.2	

		c:o.18		–	down	4.13.3.2	
		c:m		–	high	4.13.3.2.1	
		c:o.21		–	integer	4.13.3.2.1.1	
		c:o.21		–	real	4.13.3.2.1.2	
		c:m		–	low	4.13.3.2.2	
		c:o.22		–	integer	4.13.3.2.2.1	
		c:o.22		–	real	4.13.3.2.2.2	
		c:o		–	armTime	4.13.4	
		o		{smi2AttributeID 30}	trendIndication	4.14	
		o		{smi2AttributeID 6}	additionalInformation	5.1	5
		c:m		–	identifier	5.1.1	
		c:m		–	significance	5.1.2	
		c:m		–	information	5.1.3	
		o		{smi2AttributeID 7}	additionalText	5.2	
		o		{smi2AttributeID 8}	attributeIdentifierList	5.3	
		c:o.23		–	globalForm	5.3.1	
		c:o.23		–	localForm	5.3.2	
		o		{smi2AttributeID 12}	correlatedNotifications	5.4	
		c:m		–	correlatedNotifications	5.4.1	
		c:o		–	sourceObjectInst	5.4.2	
		c:o.24		–	distinguishedName	5.4.2.1	
		c:m		–	AttributeType	5.4.2.1.1	
		c:m		–	AttributeValue	5.4.2.1.2	
		c:o.24		–	nonSpecificForm	5.4.2.2	
		c:o.24		–	localDistinguishedName	5.4.2.3	
		c:m		–	AttributeType	5.4.2.3.1	
		c:m		–	AttributeValue	5.4.2.3.2	
		o		{smi2AttributeID 16}	notificationIdentifier	5.5	
		o		{smi2AttributeID 26}	sourceIndicator	5.6	
		m		{smi2AttributeID 28}	stateChangeDefinition	5.7	
		m		–	attributeID	5.7.1	
		c:o.25		–	globalForm	5.7.1.1	
		c:o.25		–	localForm	5.7.1.2	
		o		–	oldAttributeValue	5.7.2	
		m		–	newAttributeValue	5.7.3	

ح-۱۱-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۱۲ بیانیه انطباق با کلاس شیء cmipCS

جدول ح-۵۷- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	cmipCS	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx18(18)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۵۷ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۵۸ تکمیل کند.

جدول ح-۵۸- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۱۲-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۵۹ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۵۹- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-57/1b) then m else –						

ح-۱۲-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۶۰ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی شده را پشتیبانی کند.

جدول ح-۶۰- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	administrativeState	{smi2AttributeID 31}		m		m		m	
2	aetitle	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx20(20)}		–		m		x	
3	allomorphs	{smi2AttributeID 50}		x		c1		x	
4	commandSequencerId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx2(2)}		–		m		x	
4	nameBinding	{smi2AttributeID 63}		–		m		x	
5	objectClass	{smi2AttributeID 65}		–		m		x	
6	operationalState	{smi2AttributeID 35}		–		m		x	
7	packages	{smi2AttributeID 66}		–		m		x	

جدول ح-۶۰- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
1	x		x		x		
2	x		x		x		
3	x		x		x		
4	x		x		x		
5	x		x		x		
6	x		x		x		
7	x		x		x		
8	x		x		x		
c1: if not (H-57/1b) then m else –							

ح-۱۲-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۱۲-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۱۲-۵ اعلان‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» از جدول ح-۶۱ بیان کند که آیا اعلان‌های مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی را در قالب مدهای تأییدشده و تأییدنشده نشان دهد.

جدول ح-۶۱- MOCS – پشتیبانی از اعلان

						پشتیبانی		
ردیف	برچسب قالب نوع اعلان	مقدار شناسه شیء برای نوع اعلان	قیود و مقادیر	وضعیت	تأیید شده	تأیید نشده	اطلاعات افزوده
1	objectCreation }	{smi2Notification 6		m			
2	objectDeletion	{smi2Notification 7}		m			
3	stateChange	{smi2Notification 14}		m			

جدول ح-۶۱- (ادامه)

ردیف	زیرردیف	برچسب نام فیلد اعلان	مقدار شناسه شیء نوع صفت مرتبط با فیلد	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	1.1	additionalInformation	{smi2AttributeID 6}		o		
	1.1.1	identifier	–		c:m		
	1.1.2	significance	–		c:m		
	1.1.3	information	–		c:m		
	1.2	additionalText	{smi2AttributeID 7}		o		
	1.3	attributeList	{smi2AttributeID 9}		o		
	1.3.1	attributeId	–		c:m		
	1.3.1.1	globalForm	–		c:o.1		
	1.3.1.2	localForm	–		c:o.1		
	1.3.2	attributeValue	–		c:m		
	1.4	correlatedNotifications	{smi2AttributeID 12}		o		
	1.4.1	correlatedNotifications	–		c:m		
	1.4.2	sourceObjectInst	–		c:o		
	1.4.2.1	distinguishedName	–		c:o.2		
	1.4.2.1.1	AttributeType	–		c:m		

		c:m		–	AttributeValue	1.4.2.1.2	
		c:o.2		–	nonSpecificForm	1.4.2.2	
		c:o.2		–	localDistinguishedName	1.4.2.3	
		c:m		–	AttributeType	1.4.2.3.1	
		c:m		–	AttributeValue	1.4.2.3.2	
		o		{smi2AttributeID 16}	notificationIdentifier	1.5	
		o		{smi2AttributeID 26}	sourceIndicator	1.6	
		o		{smi2AttributeID 6}	additionalInformation	2.1	2
		c:m		–	identifier	2.1.1	
		c:m		–	significance	2.1.2	
		c:m		–	information	2.1.3	
		o		{smi2AttributeID 7}	additionalText	2.2	
		o		{smi2AttributeID 9}	attributeList	2.3	
		c:m		–	attributeId	2.3.1	
		c:o.3		–	globalForm	2.3.1.1	
		c:o.3		–	localForm	2.3.1.2	
		c:m		–	attributeValue	2.3.2	
		o		{smi2AttributeID 12}	correlatedNotifications	2.4	
		c:m		–	correlatedNotifications	2.4.1	
		c:o		–	sourceObjectInst	2.4.2	
		c:o.4		–	distinguishedName	2.4.2.1	
		c:m		–	AttributeType	2.4.2.1.1	
		c:m		–	AttributeValue	2.4.2.1.2	
		c:o.4		–	nonSpecificForm	2.4.2.2	
		c:o.4		–	localDistinguishedName	2.4.2.3	
		c:m		–	AttributeType	2.4.2.3.1	
		c:m		–	AttributeValue	2.4.2.3.2	
		o		{smi2AttributeID 16}	notificationIdentifier	2.5	
		o		{smi2AttributeID 26}	sourceIndicator	2.6	
		o		{smi2AttributeID 6}	additionalInformation	3.1	3
		c:m		–	identifier	3.1.1	
		c:m		–	significance	3.1.2	
		c:m		–	information	3.1.3	
		o		{smi2AttributeID 7}	additionalText	3.2	
		o		{smi2AttributeID 8}	attributeIdentifierList	3.3	
		c:o.5		–	globalForm	3.3.1	
		c:o.5		–	localForm	3.3.2	
		o		{smi2AttributeID 12}	correlatedNotifications	3.4	
		c:m		–	correlatedNotifications	3.4.1	
		c:o		–	sourceObjectInst	3.4.2	
		c:o.6		–	distinguishedName	3.4.2.1	
		c:m		–	AttributeType	3.4.2.1.1	
		c:m		–	AttributeValue	3.4.2.1.2	
		c:o.6		–	nonSpecificForm	3.4.2.2	
		c:o.6		–	localDistinguishedName	3.4.2.3	
		c:m		–	AttributeType	3.4.2.3.1	

		c:m		–	AttributeValue	3.4.2.3.2	
		o		{smi2AttributeID 16}	notificationIdentifier	3.5	
		o		{smi2AttributeID 26}	sourceIndicator	3.6	
		m		{smi2AttributeID 28}	stateChangeDefinition	3.7	
		m		–	attributeID	3.7.1	
		c:o.7		–	globalForm	3.7.1.1	
		c:o.7		–	localForm	3.7.1.2	
		o		–	oldAttributeValue	3.7.2	
		m		–	newAttributeValue	3.7.3	

ح-۱۲-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۱۳ بیانیه انطباق با کلاس شیء cmisScript

جدول ح-۶۲- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	cmisScript	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx12(12)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۶۲ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۶۳ تکمیل کند.

جدول ح-۶۳- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۱۳-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۶۴ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۶۴- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		c2		
c1: if not (H-62/1b) then m else – c2: if H-64/1 then m else –						

ح-۱۳-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۶۵ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۶۵- MOCS – پشتیبانی از صفت

			تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	administrativeState	{smi2Attribute ID 31}		m		m		m	
2	allomorphs	{smi2Attribute ID 50}		x		c1		x	
3	executionResultType	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx3(3)}		–		m		x	
4	nameBinding	{smi2Attribute ID 63}		–		m		x	
5	objectClass	{smi2Attribute ID 65}		–		m		x	
6	packages	{smi2Attribute ID 66}		–		m		x	
7	scriptId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx5(5)}		–		m		x	

جدول ح-۶۵- (نهایی)

		اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده

۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
۵	x		x		x		
۶	x		x		x		
۷	x		x		x		
c1: if not(H-62/1b) then m else – c2: if H-64/2 then m else –							

ح-۱۳-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۱۳-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۱۳-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۱۳-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۱۴ بیانیه انطباق با کلاس شیء getCmisScript

جدول ح-۶۶- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	getCmisScript	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx13(13)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۶۶ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۶۷ تکمیل کند.

جدول ح-۶۷- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۱۴-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۶۸ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۶۸- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	allomorphicPackage	{smi2Package 17}		c1		
2	packagesPackage	{smi2Package 16}		c2		
c1: if not (H-66/1b) then m else – c2: if H-68/1 then m else –						

ح-۱۴-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۶۹ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۶۹- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	administrativeState	{smi2Attribute ID 31}		m		m		m	
2	allomorphs	{smi2Attribute ID 50}		x		c1		x	
3	attributeIdentifierList	{smi2Attribute ID 8}		m		m		m	
4	baseManagedObjectId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx13(13)}		–		m		x	
5	executionResultType	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx3(3)}		–		m		x	
6	filter	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7)		m		m		m	

							xx15(15)}		
	x		m		–		{smi2Attribute ID 63}	nameBinding	7
	x		m		–		{smi2Attribute ID 65}	objectClass	8
	x		m		–		{smi2Attribute ID 66}	packages	9
	m		m		m		{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx14(14)}	scope	10
	x		m		–		{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx5(5)}	scriptId	11
	m		m		m		{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx16(16))}	synchronization	12

جدول ح-۶۹- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
۵	x		x		x		
۶	x		x		x		
۷	x		x		x		
۸	x		x		x		
۹	x		x		x		
۱۰	x		x		x		
۱۱	x		x		x		
۱۲	x		x		x		
c1: if not (H-66/1b) then m else – c2: if H-68/2 then m else –							

ح-۱۴-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۱۴-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۱۴-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۱۴-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۱۵ بیانیه انطباق با کلاس شیء setCmisScript

جدول ح-۷۰- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
1	setCmisScript	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx14(14)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۷۰ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۷۱ تکمیل کند.

جدول ح-۷۱- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
1			
2			

ح-۱۵-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۷۲ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۷۲- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	allomorphicPackage	{smi2Package 17}		c1		
2	packagesPackage	{smi2Package 16}		c2		
c1: if not (H-70/1b) then m else – c2: if H-72/1 then m else –						

ح-۱۵-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۷۳ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۷۳- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	administrativeState	{smi2AttributeID 31}		m		m		m	
2	allomorphs	{smi2AttributeID 50}		x		c1		x	
3	baseManagedObjectId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx13(13)}		–		m		x	
4	executionResultType	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx3(3)}		–		m		x	
5	filter	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx15(15)}		m		m		m	
6	modificationList	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx17(17)}		m		m		m	
7	nameBinding	{smi2AttributeID 63}		–		m		x	
8	objectClass	{smi2AttributeID 65}		–		m		x	
9	packages	{smi2AttributeID 66}		–		m		x	
10	scope	{joint-iso-itu-t ms(9) function(2)		m		m		m	

							part21(21) attribute(7) xx14(14)}		
	x		m		–		{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx5(5)}	scriptId	11
	m		m		m		{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx16(16))}	synchronization	12

جدول ح-۷۳- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	**وضعیت**	**پشتیبانی**	**وضعیت**	**پشتیبانی**	**وضعیت**	**پشتیبانی**	**اطلاعات افزوده**
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
۵	x		x		x		
۶	x		x		x		
۷	x		x		x		
۸	x		x		x		
۹	x		x		x		
۱۰	x		x		x		
۱۱	x		x		x		
۱۲	x		x		x		
c1: if not (H-70/1b) then m else – c2: if H-72/2 then m else –							

ح-۱۵-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۱۵-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۱۵-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۱۵-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۱۶ بیانیه انطباق با کلاس شیء actionCmisScript

جدول ح-۷۴- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	actionCmisScript	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx15(15)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۷۴ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۷۵ تکمیل کند.

جدول ح-۷۵- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۱۶-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۷۶ بیان کند که آیا بسته‌های شرطی مشخص‌شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۷۶- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-74/1b) then m else – c2: if H-76/1 then m else –						

ح-۱۶-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۷۷ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۷۷- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	administrativeState	{smi2AttributeID 31}		m		m		m	
2	allomorphs	{smi2AttributeID 50}		x		c1		x	
3	baseManagedObjectId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx13(13)}		–		m		x	
4	executionResultType	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx3(3)}		–		m		x	
5	filter	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx15(15)}		m		m		m	
6	nameBinding	{smi2AttributeID 63}		–		m		x	
7	objectClass	{smi2AttributeID 65}		–		m		x	
8	packages	{smi2AttributeID 66}		–		m		x	
9	scope	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx14(14)}		m		m		m	
10	scriptId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx5(5)}		–		m		x	
11	synchronization	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx16(16))}		m		m		m	

جدول ح-۷۷- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
1	x		x		x		
2	x		x		x		
3	x		x		x		
4	x		x		x		
5	x		x		x		
6	x		x		x		
7	x		x		x		
8	x		x		x		
9	x		x		x		
10	x		x		x		
11	x		x		x		
c1: if not (H-74/1b) then m else – c2: if H-76/2 then m else –							

ح-۱۶-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۱۶-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۱۶-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۱۶-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۱۷ بیانیه انطباق با کلاس شیء createCmisScript

جدول ح-۷۸- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
1	createCmisScript	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx16(16)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۷۸ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۷۹ تکمیل کند.

جدول ح-۷۹- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۱۷-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۸۰ بیان کند که آیا بسته‌های شرطی مشخص شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۸۰- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
1	allomorphicPackage	{smi2Package 17}		c1		
2	managedObjectInstancePackage	{joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx16(16)}		c2		
3	packagesPackage	{smi2Package 16}		c3		
4	referenceObjectInstancePackage	{joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx18(18)}		c4		
5	superiourObjectInstancePackage	{joint-iso-itu-t ms(9) function(2) part21(21) package(4) xx17(17)}		c5		

c1: if not (H-78/1b) then m else –
c2: if “the superiourObjectInstancePackage is not present” then m else –
c3: if H-80/1 or H-80/2 or H-80/4 or H-80/5 then m else –
c4: if “the manager has the specified value” then m else –
c5: if “the managedObjectInstance Package is not present” then m else –

ح-۱۷-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۸۱ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۸۱- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	administrativeState	{smi2AttributeID 31}		m		m		m	
2	allomorphs	{smi2AttributeID 50}		x		c1		x	
3	attributeList	{smi2AttributeID 9}		m		m		m	
4	executionResultType	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx3(3)}		–		m		x	
5	managedObjectInstance	{smi2AttributeID 61}		c2		c2		c2	
6	nameBinding	{smi2AttributeID 63}		–		m		x	
7	objectClass	{smi2AttributeID 65}		–		m		x	
8	packages	{smi2AttributeID 66}		–		m		x	
9	referenceObjectInstance	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx19(19)}		c4		c4		c4	
10	scriptId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx5(5)}		–		m		x	
11	superiourObjectInstance	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx18(18)}		c5		c5		c5	

جدول ح-۸۱- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی	اطلاعات افزوده
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
۵	x		x		x		
۶	x		x		x		
۷	x		x		x		
۸	x		x		x		
۹	x		x		x		
۱۰	x		x		x		
۱۱	x		x		x		

c1: if not (H-78/1b) then m else –
c2: if H-80/2 then m else –
c3: if H-80/3 then m else –
c4: if H-80/4 then m else –
c5: if H-80/5 then m else –

ح-۱۷-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۱۷-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۱۷-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۱۷-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.

ح-۱۸ بیانیه انطباق با کلاس شیء deleteCmisScript

جدول ح-۸۲- MOCS – پشتیبانی از کلاس شیء مدیریت‌شده

ردیف	برچسب قالب کلاس شیء مدیریت‌شده	مقدار شناسه شیء برای کلاس	پشتیبانی برای همه ویژگی‌های اجباری	آیا کلاس واقعی همانند کلاس شیء مدیریت‌شده‌ای است که ادعای انطباق با آن صورت گرفته است ؟ (Y/N)
۱	deleteCmisScript	{joint-iso-itu-t ms(9) function(2) part21(21) managedObjectClass(3) xx17(17)}		

اگر پاسخ پرسش کلاس واقعی پشتیبانی از کلاس شیء مدیریت‌شده در جدول ح-۸۲ منفی باشد، تأمین‌کننده‌ی پیاده‌سازی باید پشتیبانی کلاس واقعی را در جدول ح-۸۳ تکمیل کند.

جدول ح-۸۳- MOCS – پشتیبانی کلاس واقعی

ردیف	برچسب قالب کلاس شیء مدیریت شده واقعی	مقدار شناسه شیء برای کلاس واقعی	اطلاعات افزوده
۱			
۲			

ح-۱۸-۱ بسته‌ها

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۸۴ بیان کند که آیا بسته‌های شرطی مشخص شده به‌وسیله این کلاس، به‌وسیله نمونه‌ای از این کلاس مورد پشتیبانی هستند یا خیر.

جدول ح-۸۴- MOCS – پشتیبانی از بسته

ردیف	برچسب قالب بسته	مقدار شناسه شیء برای بسته	قیود و مقادیر	وضعیت	پشتیبانی	اطلاعات افزوده
۱	allomorphicPackage	{smi2Package 17}		c1		
۲	packagesPackage	{smi2Package 16}		m		
c1: if not (H-82/1b) then m else – c2: if H-84/1 then m else –						

ح-۱۸-۲ صفات

تأمین‌کننده‌ی پیاده‌سازی باید در ستون‌های «پشتیبانی» و «اطلاعات افزوده» در جدول ح-۸۵ بیان کند آیا صفات مشخص‌شده به‌وسیله همه بسته‌هایی که در یک شیء مدیریت‌شده این کلاس نمونه‌سازی شده‌اند، پشتیبانی می‌شوند یا خیر. تأمین‌کننده‌ی پیاده‌سازی باید هر یک از عملیات و هر یک از صفات پشتیبانی‌شده را پشتیبانی کند.

جدول ح-۸۵- MOCS – پشتیبانی از صفت

				تنظیم با ایجاد		گرفتن		تعویض	
ردیف	برچسب قالب صفت	مقدار شناسه شیء برای صفت	قیود و مقادیر	وضعیت	پشتیبانی	وضعیت	پشتیبانی	وضعیت	پشتیبانی
1	administrativeState	{smi2AttributeID 31}		m		m		m	
2	allomorphs	{smi2AttributeID 50}		x		c1		x	
3	baseManagedObjectId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx13(13)}		–		m		x	
4	executionResultType	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx3(3)}		–		m		x	
5	filter	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx15(15)}}		m		m		m	
6	nameBinding	{smi2AttributeID 63}		–		m		x	
7	objectClass	{smi2AttributeID 65}		–		m		x	
8	packages	{smi2AttributeID 66}		–		c2		x	
9	scope	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx14(14)}		m		m		m	
10	scriptId	{joint-iso-itu-t ms(9) function(2) part21(21) attribute(7) xx5(5)}		–		m		x	
11	synchronization	{joint-iso-itu-t ms(9) function(2)		m		m		m	

							part21(21) attribute(7) xx16(16)}		

جدول ح-۸۵- (نهایی)

	اضافه کردن		حذف کردن		تنظیم به پیش‌فرض		
ردیف	**وضعیت**	**پشتیبانی**	**وضعیت**	**پشتیبانی**	**وضعیت**	**پشتیبانی**	**اطلاعات افزوده**
۱	x		x		x		
۲	x		x		x		
۳	x		x		x		
۴	x		x		x		
۵	x		x		x		
۶	x		x		x		
۷	x		x		x		
۸	x		x		x		
۹	x		x		x		
۱۰	x		x		x		
۱۱	x		x		x		
c1: if not (H-82/1b) then m else – c2: if H-84/2 then m else –							

ح-۱۸-۳ گروه‌های صفت

هیچ گروه صفتی برای کلاس شیء مدیریت‌شده تعریف نشده است.

ح-۱۸-۴ عمل‌ها

هیچ عملی برای این کلاس شیء تعریف نشده است.

ح-۱۸-۵ اعلان‌ها

هیچ اعلانی برای این کلاس شیء تعریف نشده است.

ح-۱۸-۶ پارامترها

هیچ پارامتری برای این کلاس شیء تعریف نشده است.