IPA Software Engineering Center
Information-technology Promotion Agency, Japan

形式手法適用調査

調査概要資料

2010年7月29日

独立行政法人　情報処理推進機構
ソフトウェア・エンジニアリング・センター

調査対象・項目と目次

SEC
Software Engineering
for Mo・No・Zu・Ku・Ri

調査対象

- 欧州
  - ツールベンダ
    サービスプロバイダ
  - 形式手法ユーザ
  - 研究者（大学教授）
- 文献／Web
- 米国
  - 研究者
    ツールベンダ等

調査項目

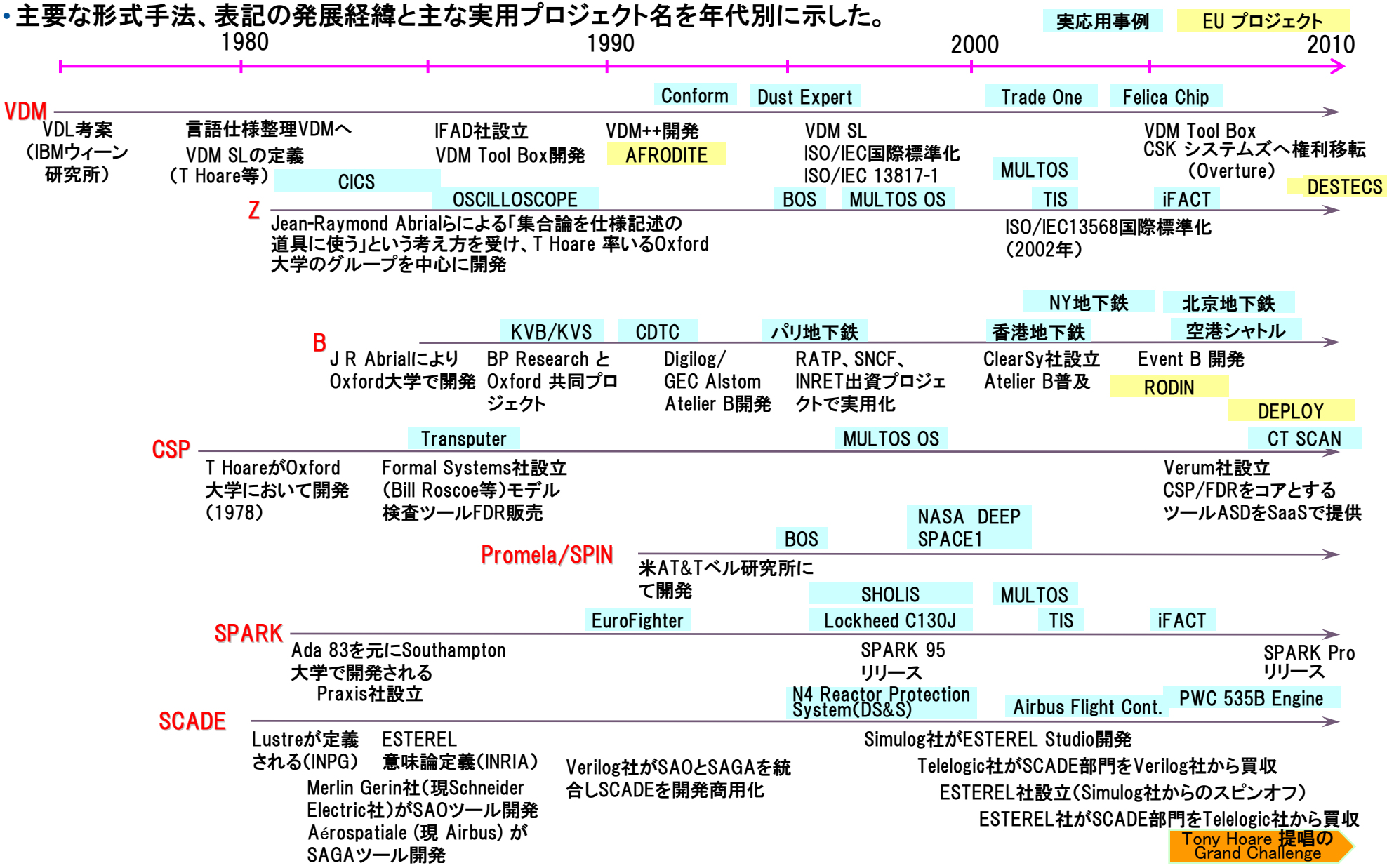
- 実応用事例
  （事例データベース）
- 手法、表記
  ツール／ベンダ
  要求スキル
  教育／トレーニング
  国際標準
  調達基準

目次

形式手法発展と実応用の経緯

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

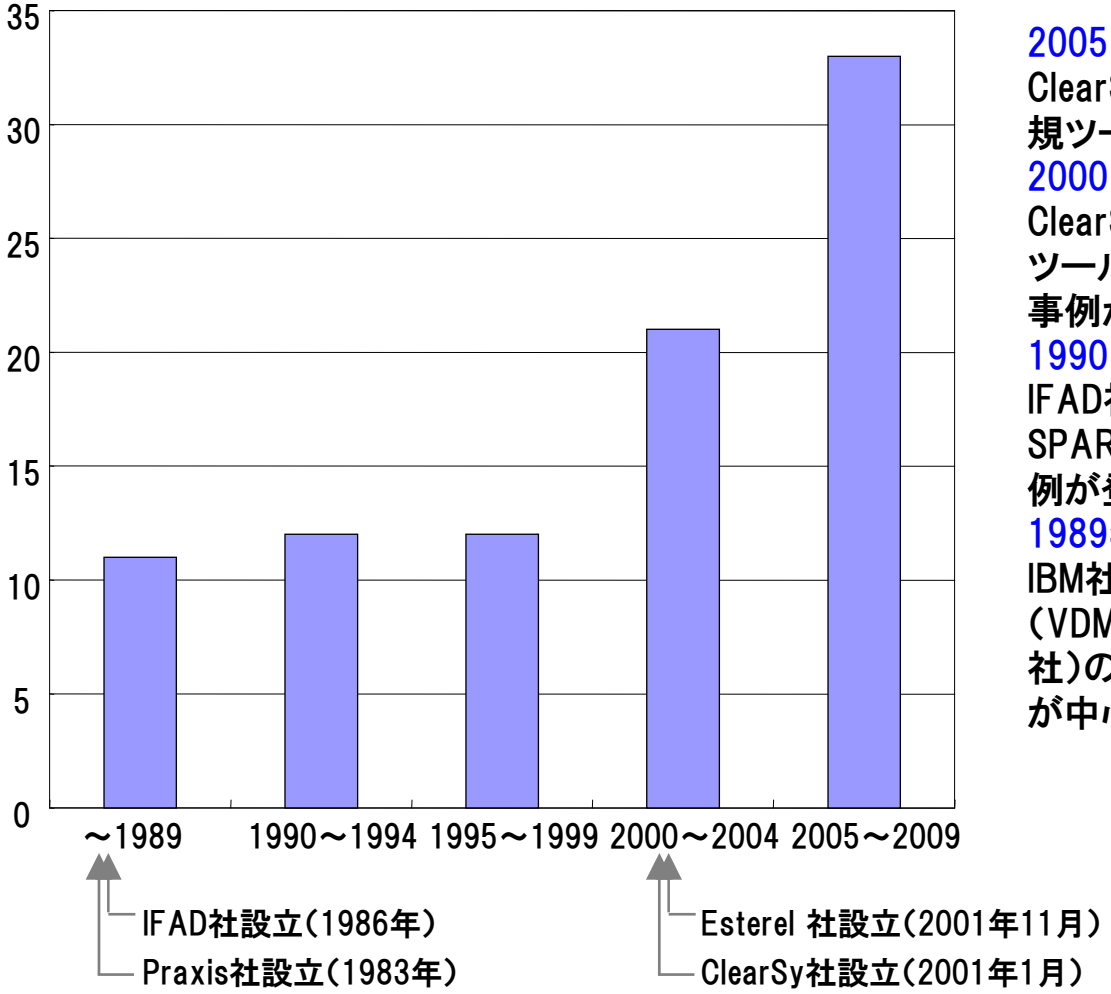
- 主要な形式手法、表記の発展経緯と主な実用プロジェクト名を年代別に示した。



形式手法実応用プロジェクト:公開事例の年代別出現数

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- 形式手法を利用した実応用開発を、論文、ツールベンダの事例紹介、コンファレンスでの発表資料から収集し、その結果を開発を開始した年代別に分類した。プロジェクトの対象は実応用プロジェクトとし、研究開発レベルのプロジェクトは除いている。



2005～2009年
ClearSy社、Esterel社の実績に加えVerum社のような新規ツールベンダの参入により事例は増加傾向。

2000～2004年
ClearSy社ツールの鉄道システムへの普及、Esterel社ツールの原子力、航空関連への普及等により、実応用事例が増加。

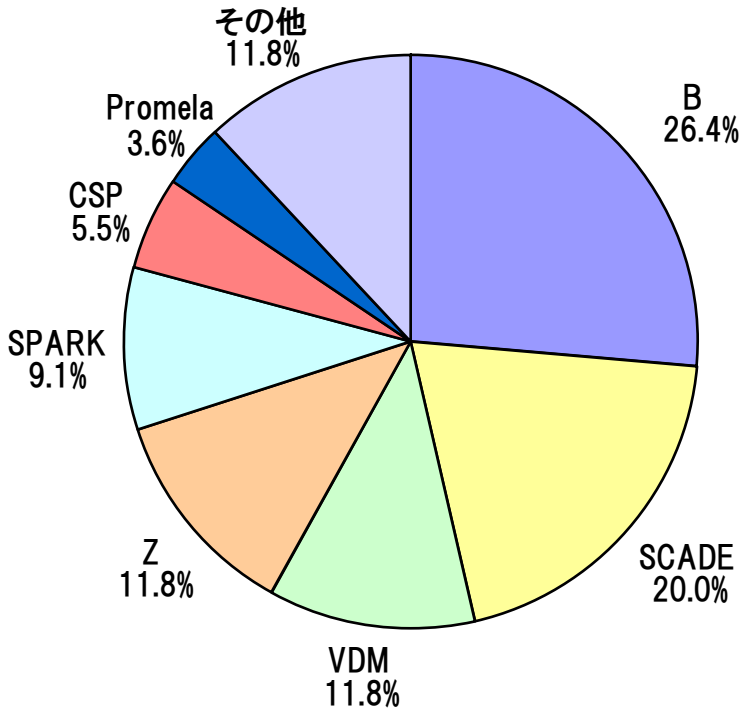
1990～1999年
IFAD社によるVDM採用のプロジェクトやPraxis社によるSPARK(航空、防衛)の実績等ツールベンダによる事例が登場。

1989年以前
IBM社のCICS;Customer Information Control System(VDM)、GEC Alstom(現Alstom社)/MATRA(現STS社)の鉄道システム(B)等、システムメーカによる事例が中心。

形式手法実応用プロジェクト:公開事例の表記別比率

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- 形式手法を利用した実応用開発を、論文、ツールベンダの事例紹介、コンファレンスでの発表資料等から収集し、その結果を採用した表記別に分類した。プロジェクトの対象は実応用プロジェクトとし、研究レベルのプロジェクトは除いている。



B
STS社(Siemens Transport System)、Alstom社等、フランスの鉄道システムを中心に、Bを採用した形式手法実応用例が多くある。

SCADE
航空宇宙(AirBus)、原子力、鉄道などの事例が多い。

VDM
日本での事例として、Trade OneやFelica Chipの開発への適用がある。

Z
仕様記述表記に使われ、ZおよびSPARKの組合せで詳細化をする例が見られる。

SPARK
航空管制システム(iFACTS)や航空機(民生、軍事)、セキュリティ(スマートカードや認証)の事例がある。

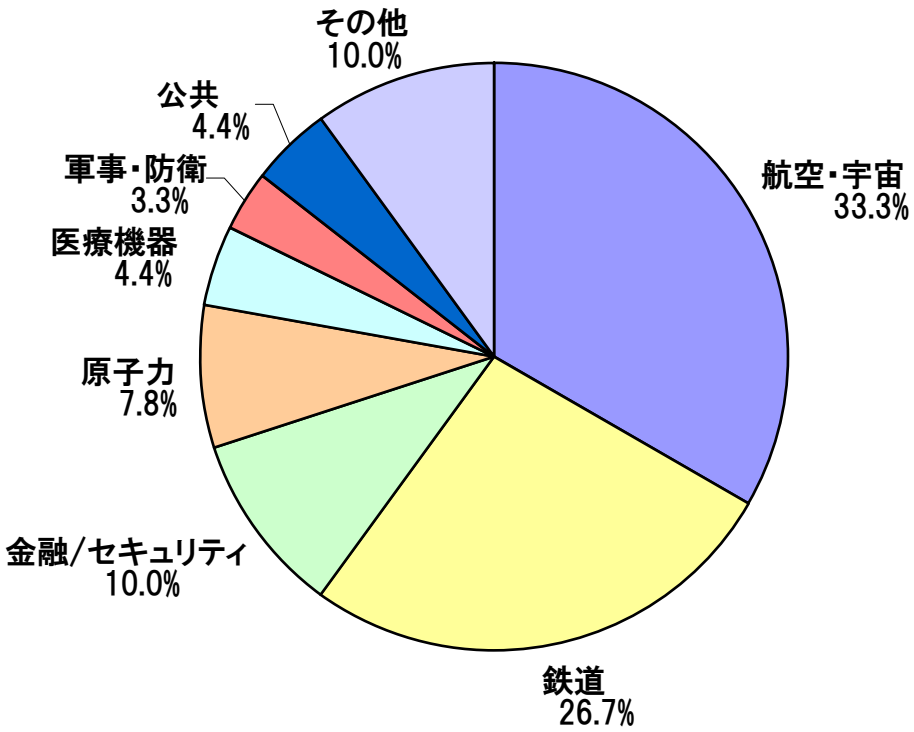
CSP
最近の例としてVerum社(CSPを利用)のツールを採用した例(Philips)などがある。

Promela
SPINの言語としてモデル検査に用いられる例(オランダ防潮提開閉意志決定システム)がある。

形式手法実応用プロジェクト:公開事例の業界別比率

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- 形式手法を利用した実応用開発を、論文、ツールベンダの事例紹介、コンファレンスでの発表資料から収集し、その結果を業界別に分類した。プロジェクトの対象は実応用プロジェクトとし、研究開発レベルのプロジェクトは除いている。航空・宇宙、鉄道などの大規模製品への利用が多い。OSやスマートカード、医療機器等の部品、サブシステムでの利用もある。



航空
AirBus社でのSCADEの採用、航空管制システムの開発におけるVDMやSPARKの採用などがある。

鉄道
フランスのSTS社(Siemens Transport System)、GEC Alstom社などによる鉄道システムにおいて、Bを採用した形式手法実応用例が多くある。

金融/セキュリティ
銀行システムの開発、スマートカードなどのセキュリティでの形式手法の採用がある。

原子力
Schneider Electric社でのSCADE採用等の事例がある

医療機器
Philips社の医療機器においてオランダのツールベンダであるVerum社のツールを利用した開発例がある。

軍事・防衛
英国の防衛システムの開発において形式手法が採用されており、SPARK(Praxis社)による開発例が多くある。

公共
オランダの防潮可動堤開閉意志決定システムで、Promela/SPINを使用した例がある。

その他
OpenComRTOS(Altreonic社/ベルギ)、フラッシュメモリデバイスドライバ(Samsung)などがある。

実開発で使用されている主な形式手法、表記(1/2)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

・主要な形式手法、表記について示す。

記法/手法	VDM手法	Z表記	B手法	Event B
開発者／経緯	■IBMウィーン研究所で開発(1960-1970年代) ■PL/Iコンパイラの正しさを検証するプロジェクトがきっかけ ■Tony Hoareなどが中心的役割を果たし、PL/I言語定義のメタ言語としてVDLを考案 ■言語仕様が整理されてVDMとなる ■その後研究拠点はデンマークと英国へ移る ■VDM-SL(1996年ISO IEC 13817標準化) 参照:NII技術レポート	■Jean-Raymond Abrialらによる「集合論を仕様記述の道具に使う」という考え方を受け、Tony Hoare率いるOxford大学のグループを中心に開発された(1980年代) ■ZF集合論から名前を取って命名された ■ISO/IEC13568国際標準化(2002年)され、OSIネットワーク管理モデルの仕様記述に利用	■Jean-Raymond AbrialがOxford大学でZ-表記に引き続き概念を開発(1985) ■BP出資の研究プロジェクト(BP IT 部門とOxford)(1985-1988) ■Digilog社-GEC Alstom社がAbrielの協力によりAtelier Bツールキット開発(1988-1992) ■RATP、INRET、SNCFのプロジェクトでAtelier B強化	■EU FP6プロジェクト「RODIN」で、RODINプラットフォーム(ツール)と併せて開発された。 ■その後「Deploy」に引き継がれ、ツールの開発が進められている。
特徴	■抽象的なデータ定義と操作仕様から始めて、順次、データ構造を具体化し、アルゴリズムに展開することでプログラムを作成 ■モデルベースの記述に適するが、タイムベースのシステムの記述には向かない ■VDM++(オブジェクト指向版の記述言語／ESPRIT計画のAFRODITEプロジェクトで開発)	■公理的集合論、ラムダ計算、一階述語論理で使 われる数学的記法に基づいて、プログラム仕様と意図する振る舞いをシステムの状態と操作として記述する ■Z++、Object-Z、TCOZ(オブジェクト指向の拡張版) 参照:形式手法概説/レンタコーチ	■詳細化(リファイメント)を中心とする段階的な詳細化の技法によって仕様からプログラム導出までをカバーする 参照:NII技術レポート ■Event B(RODINプロジェクトで開発)	■Bメソッドの詳細化(リファインメント)技法に基づくCorrectness by Constructionの考え方を開発上流のモデリングまで広げた形式手法 ■対象を並行システムとして表現するためにガードコマンドの方法を採用
ツール	■VDM Tool Kit(デンマークIFAD社からCSKシステムズへ権利移転)	■Community Z Tools(オープンソース) ■Proof Power(仕様記述解析) ■Isabelle/HOL-Z(仕様記述解析)	■Atelier B(ClearSy) ■B4Free(フリーウェア) ■B-Tool及びB-Toolkit(B-Core) 注)開発者が療養中であったため、B-Core社は現在積極的な活動をしていない ■EdithB(STS社内製詳細化ツール) ■BART(ClesrSy、詳細化ツール)	■RODIN Platform ■ProB
適用事例	■Trade One (マル優サブシステム、オプションサブシステム) ■Felica Chip ■Dust Expert	■CICS(IBM社のOLTPモニタのAPI仕様記述に適用) ■iFACT(航空管制システム)(Praxis) Praxis社によるZとSPARKを利用したプロジェクトが多くある	■パリ地下鉄14号線(STS) ■NY地下鉄(STS) ■北京地下鉄(Alstom) STSやAlstomが実施する鉄道システムプロジェクトが多くある	EUプロジェクトのRODINやDEPLOYでEvent Bを使った実証プロジェクトの事例がある ■CDIS航空管制システムの再構築(RODIN) ■SanJuan鉄道システム(DEPLOY)

実開発で使用されている主な形式手法、表記(2/2)

SEC
Software Engineering
for Mo・No・Zu・Ku・Ri

・主要な形式手法、表記について示す。

記法/手法	CSP(Communicating Sequential Processes)手法	Promela表記	SPARK言語	SCADE言語
開発者／経緯	■Tony Hoare氏がOxford大学において開発(1978年)	■Gerard J. Holzmann等によりベル研究所で開発された(1980年)	■Ada 83を元にSouthampton大学で英国防衛省の支援を受け開発された(1983年) ■その後Program Validation社、さらにはPraxis HIS社により拡張、改善された。	■Lustre表記(Lustre V3)を拡張した言語LUSTREは、1980年代にリアルタイム制御ソフトウェアを記述するコンピュータ言語としてINPGにより定義された。 ■現同社Chief Scientist G. BerryらがEsterel意味論定義(1984年) ■EsterelとLustreが統合され現在のSCADE表記となる
特徴	■平行動作するプロセス群の振る舞いを記述する手法	■モデル検証ツールであるSPINにおいて対象となるモデル記述する表記法 モデル記述と検査に適している。	■Adaのサブセット SPARKは、煩わしい言語機能を取り除き、機能を制限する。その代りにアノテーションにより付加情報を提供できる。 ■Zで記述された機能仕様がSPARKにより詳細化される事例が多くある(Praxisの事例)	■記述したモデルの形式検証 ■IEC61508認証を受けた自動コード生成機能 ■時間に同期するリアルタイムシ　ステム向け記述(リアルタイムシステム向けの同期型プログラミング言語)
ツール	■ASDスィート(Verum) オランダのVerum社がCSPによる形式手法ツールを開発販売(モデル検証にFDRを利用) ■FDR2 英国のFormal SystemsがFDR2というモデル検証ツールを提供	■SPIN	■SPARK toolset/SPARK Pro ・SPARK Examiner(静的チェック) ・SPARK Simplifier(自動数理証明)等	■SCADE suite 6.1 ・SCADE Suite LabVIEWゲートウェイ ・SCADE Suite Timing Verifier ・SCADE Suite Stack Verifier ・SCADE Suite KCG 6.0.1 ・SCADE Display 6.1
適用事例	■マイクロプロセッサ/Transputer(INMOS) ■CTスキャナー(Philips)	■オランダ防潮可動堤開閉意思決定システム(CMG Den Hagg社) ■DEEP SPACE1(NASA) ■フラッシュメモリ(サムソン)	■SHOLIS(ヘリコプタ着陸システム) ■「Tokeneer ID Station (TIS) 」(バイオメトリクスID認証ツール) ■iFACT(航空管制システム)(Praxis) Praxis社によるZとSPARKを利用したプロジェクトが多くある	■Airbus社の航空機(A340-500/600、A380) ■サンパウロ地下鉄プラットフォームドア(POSCON) ■原子力制御システム

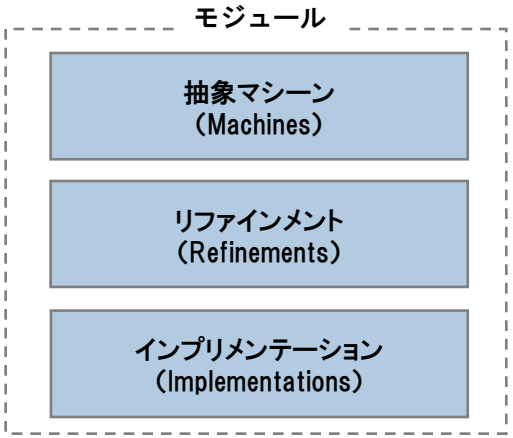
実開発で使用されている主なツールとその機能(1/3)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

・主要な形式手法のツールについて示す。

ツール名	Atelier B	Edith B	BART
開発組織	ClearSy	STS	ClearSy
提供形態	商用	社内利用	商用
手法/表記	B	B	B
開発経緯	■Digilog社-GEC Alstom社がAbrielの協力によりAtelier Bツールキット開発(1988-1992) ■RATP、INRET、SNCFのプロジェクトでAtelier B強化 ■Digilog社を買収したSteria社からのスピンアウトでClearSy社設立	■Bメソッドの詳細化(Refinement)プロセスの効率化のためにSTSが内製ツールとして開発 ■1999年に開発を完了した。	■フランスのGIP ANR(公益団体法人全国研究機構)による国家プロジェクトRIMEL(Incremental refinement of event models)の成果を活用
機能	■Bファイルの静的チェック(文法、タイプチェック、ビジビリティチェック等) ■証明責務の生成 ■自動とインタラクティブな証明 ■Bインプリメンテーションの従来言語(Ada等)への変換 Atelier Bのサブセットとして研究用途に無償で提供されるB4Freeツールがある。	■詳細化のルールを蓄積する「ルールベース」とインプリメンテーションを生成する「インタプリタ」から構成される。 ■詳細化は自動モードとインタラクティブモードで行われる。 ■適合するルールを発見できなかった場合や詳細化されたコードが正しくない、あるいは効率的でない場合にインタラクティブモードが使われる。 ■インタラクティブモードの結果はルールベースに蓄積される。	自動リファインメントツール(B Automatic Refinement Tool) ■マシーンとリファインメントから自動的にB0インプリメンテーションを生成する。 ■Atelier B 4.0に組込まれ、グラフィカルに操作可能

参考:Bモデルの構造



■Bモデルはモジュールにより構成され、データとデータの操作を行う。モジュールは、抽象マシーン、リファイメント、インプリメンテーションから構成される。

■マシーンはモジュールの外観を示す。

■リファインメントはモジュールの内観を示し、モジュールのふるまいを定義する。

■マシーンとリファイメントは抽象コンポーネント(抽象モデル)である。

■インプリメンテーションは、より具体的な記述であり、従来の言語に直接変換可能である。データ構造は限られている(整数、ブーリアン、配列)。

実開発で使用されている主なツールとその機能(2/3)

SEC
Software Engineering
for Mo・No・Zu・Ku・Ri

- 主要な形式手法のツールについて示す。

ツール名	Rodin Platform	ProB	SCADE suite 6.1	ASD(Verum)	FDR2
開発組織	EU RODINプロジェクト	EPSRCプロジェクト	ESTEREL	Verum	Formal Systems
提供形態	参加企業による利用	オープンソース	商用	商用(SaaS)	商用
手法/表記	Event B	B、Event B	SCADE	CSP	CSP
開発経緯	■EU FP6プロジェクト「RODIN」で、RODINプラットフォーム(ツール)と併せて開発された。 ■その後「Deploy」に引き継がれ、ツールの開発が進められている。	■英国のプロジェクトEPSRCがファンドしたABCDプロジェクトとiMocプロジェクトで開発された。. ■Southampton大学のMichael Leuschel.教授により保守されている。	■Verilog社(ソフトウェア・ベンチャー)がMerlin Gerin(現Schneider Electric傘下・ブランド名)の内製ツールSagaとAérospatiale社(現AirBus社)のツールSAOを統合し商用化(1986年) ■Telelogic社がVerilog社SCADE開発途中にVerilog/SCADEを買収(1999年3月) ■Esterel社、SCADE部門をTelelogicより買収(2001年11月)	■オランダのVerum社がCSPによる形式手法ツールを開発販売(モデル検証にFDRを利用) ■Oxford大学にてT Hoarelに師事した教授の研究室でPhdを取得したVerum社のCTOの娘とCTOが中心となり開発	■Oxford大学コンピュータサイエンス学部Bill Roscoe教授(T Hoareに師事)により開発された。 ■1989年同教授によりFormal Systems社が設立され、商用化された(学術用途の利用はフリー)
機能	■Eclipseベースの統合開発環境 ■Bメソッドの詳細化(リファインメント)技法に基づくCorrectness by Constructionの考え方を開発上流のモデリングまで広げた形式手法 ■対象を並行システムとして表現するためにガードコマンドの方法を採用	Bによる仕様記述を対象として、仕様アニメーション, モデル検査を行うための処理系 ■B 仕様記述の文法チェック ■型チェック ■対話的、もしくはランダム実行による仕様アニメーションモデル検査 ■状態遷移グラフの生成	■ソフトウェア要件管理 ■ソフトウェア開発(制御モデル設計) ■検証(シミュレーション、形式検証) ■自動Cコード生成 IEC-61508 SIL3/SIL4認証を取得し、制御モデルと自動生成されるCコードとの一致性を保証	タビュラー入力により定義されるASD制御モデル(モデルの内部構造はCSPで構成)を検証(FDRを使用)し、既存の言語へ変換する。 ■ASDモデル生成(Model Builder) ■ASDモデル検証(Model Checker) ■ASDモデルをC、C##、C++、Javaなどの言語へ変換(Code Generator)	■CSP理論に基づく並行システム検証ツール. (モデル検査器)…

実開発で使用されている主なツールとその機能(3/3)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

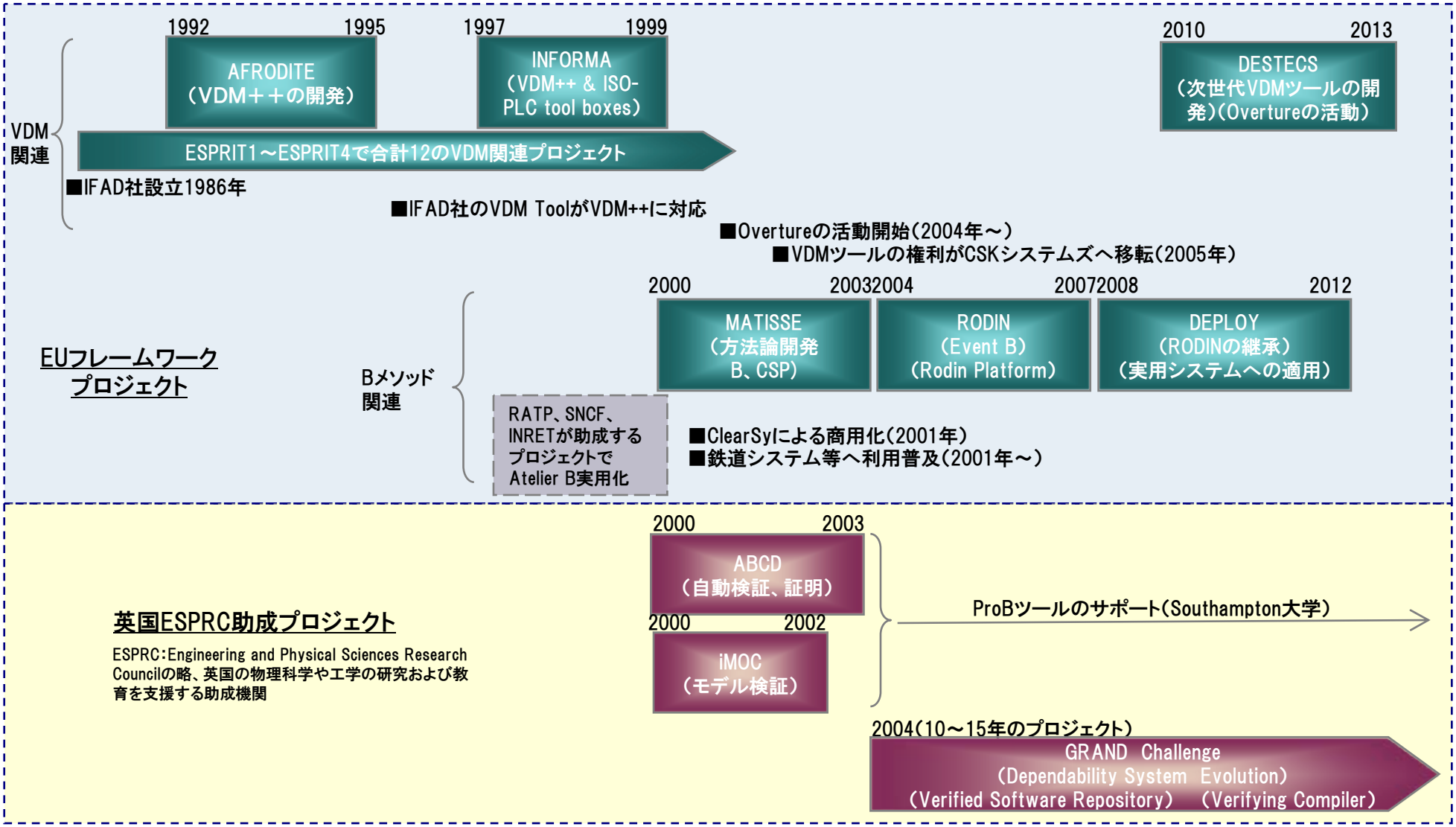
- 主要な形式手法のツールについて示す。

ツール名	SPARK Pro	SPIN	VDM Tool Kit	Overture IDE	Perfect Developer
開発組織	Praxis HIS	ベル研究所	CSKシステムス(IFAD)	開発コミュニティ	Escher
提供形態	商用	オープンソース	商用	オープンソース	商用
手法/表記	SPARK	Promela	VDM	VDM	Perfect Languege
開発経緯	■Ada 83を元にSouthampton大学で英国防衛省の支援を受け開発された(1983年) ■その後Program Validation社、さらにはPraxis HIS社により拡張、改善された。	■Gerard J. Holzmann等によりベル研究所で開発された(1980年)	■デンマークのIFAD社が1980年代に開発し、現在は、CSKシステムズが引き継いでいる。 ■国内のCSKシステムズが、VDM-SL及びVDM++に対応するVDM Toolsを販売	■Peter Larcen教授が中心となりVDMのツールの開発をするコミュニティ(グループ)であるOvertureが開発する(2005～) ■EU-FP7プロジェクトDESTECS 予算300万ユーロ 2010年1月開始予定、 期間3年間)の一部として活動が進む予定	■David Crocker氏により、1995年に設立されたEscher社(英)により開発された。
機能	■静的チェック、情報フロー分析、検証条件(VC)設定(Examiner) ■自動数理証明(検証条件に基づく検証)(Simplifier) ■証明チェック(自動数理証明で取残された証明の実行) (Proof Checker)	■分散SWシステムのモデル検査ツール ■非同期分散アルゴリズムに向く. ■検証条件はLTL(Linear Temporal Logic)で記述される。 ■モデル検査だけでなくシミュレータの機能も持ち、実行履歴を取ることができる。	■VDM構文エラーチェック(構文チェック) ■VDM仕様のデータ型整合性の静的チェック(型チェック) ■仕様実行・テストケースに基づく動的チェック(インタプリタとデバッガ) ■テストカバレッジの表示(コードカバレッジ) ■UML⇔VDM++の生成(ROSEリンク) ■VDM++からC++やJava生成(コード生成) ■インタプリタでC++とダイナミックリンク(ダイナミックリンク)	■EclipseベースのVDM++の開発プラットフォーム ■DESTECSの活動でVDMToolに代わる新たなツールを開発する予定 ■オープンソースVDMツール開発 Overture Toolset、VDM PF開発 OML(Overture Modeling Language/VDM++拡張版)	■仕様記述、仕様検証、詳細化の一連のプロセスを1種類の表記方法で行い、詳細仕様の検証し自動的に既存のプログラミング言語(C++、Java2、Ada95)に変換する。

形式手法に関連する欧州のランドマーク研究開発プロジェクト(1/5)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- VDM++、Bメソッドの形式手法やそのツールは、EUやUKの助成プロジェクトを活用し、さらなる進化をしている。



形式手法に関連する欧州のランドマーク研究開発プロジェクト(2/5)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- VDM++とそのツールは、FP3 ESPRITの「AFRODITE」プロジェクトで開発された。VPM++の開発プラットフォームを研究する「DESTECS」プロジェクト(FP7) が2010年から始まる。

AFRODITE(蘭)(ESPRIT3/FP3)

- **名称**
  - 形式手法を、技術環境において実用規模オブジェクト指向設計に適用する（Applying Formal Methods to Real-Size Object-Oriented Designs in Technical Environments）
- **期間**
  - 1992～1995年
- **予算**
  - N/A
- **主要参加組織**
  - CAP GEMINI INTERNATIONAL SUPPORT 、Instituttet for Anvendt Datateknik （IFAD）（デンマーク）、Manchester大学（英）、欧州原子力研究機構（CERN）他
- **研究内容**
  - VDM-SLをオブジェクト指向拡張したVDM++の開発
  - 形式仕様のソフトウェア/ハードウェア統合システムへの適用が、VDMの拡張版を用いて実証された。
  - VDM++の既存（非形式手法）開発手法やツールへの組込み
  - OMT CASE環境の付加モジュールとしてオブジェクトモデル技術（OMT）と統合
  - グラフィカルユーザインタフェースにより、VDM++チェッカ、コード生成、プリティプリンティング機能をサポートし、OMTモデルからVDM++が生成可能なツールへリンクされるツールセット

DESTECS(FP7)

- **名称**
  - 組込み制御ソフトウェア向け設計サポートとツール（Design Support and Tooling for Embedded Control Software）
- **組込**
  - 2010年1月開始（3年間）
- **予算**
  - 総額：3.600万ユーロ
  - 助成額： 2.700万ユーロ
- **主要参加組織**
  - Twente大学（蘭）、Newcastle大学（英）、Arhus工科大学（デンマーク）、Control lab products CLP（蘭）、Neopost（蘭）、Verhaert New Products & Services（蘭）、CHESS（蘭）
- **研究内容**
  - 組込みリアルタイム制御システムのツールや方法論
  - Peter Larsen教授（元IFAD、現Arhus工科大学教授）が中心となり活動しているOverture がDESTECSの活動の1部となり、VDM++の新たなツール、Eclipseベースのオープンツールアーキテクチャを開発する。

形式手法に関連する欧州のランドマーク研究開発プロジェクト(3/5)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

・Bメソッドの拡張であるEvent Bとツールプラットフォームが「RODIN」プロジェクトで開発され、開発成果とその活動が「DEPLOY」プロジェクトへ継承される。

RODIN(英)(FP6)

■ **名称**
- 複雑なシステムのための厳格なオープン開発環境（Rigorous Open Development Environment for Complex Systems）

■ **期間**
- 2004～2007年

■ **予算**
- 総額： 440万ユーロ（約5.3億円*）
- 助成額：317万ユーロ（約3.8億円*）

■ **期間**
- 2004～2007年

■ **主要参加組織**
- Newcastle大学 （英）、Praxis HIS（英）、ClearSy（仏）他

■ **研究内容**
- Event B表記の開発
- Eclipse ベースのRODINプラットフォームの開発
- プラグイン評価（ProB、Brama等）
- 適用事例研究
  - ✓ CDIS （航空管制ディスプレイ） （Praxis）
  - ✓ モデルドリブンアーキテクチャ（Nokia）
  - ✓ エンジン障害管理（ATEC）
  - ✓ MITA PFサブセット形式化（Nokia）他

DEPLOY(英)(FP7)

■ **名称**
- 高生産性、高信頼性の先進エンジニアリングメソッドの産業界への展開（Industrial deployment of advanced system engineering methods for high productivity and dependability）

● **期間**
- 2008～2012年

■ **予算**
- 総額：17,885,404ユーロ
- 助成額：12,059,999ユーロ

■ **主参加組織**
- Newcastle大学（英）、STS（仏）、SAP（独）、ClearSy（仏）、Systerel（仏）、Nokia（フィンランド）、Bosch（独）

■ **研究内容**
- RODINプラットフォームの継続開発
  理論コンポーネントと規則ベース証明、グラフィック言語、コード生成、要求仕様トレース、再利用（汎用開発のインスタンス化、詳細化パターン）、UML-BとEVENT Bの連係、ステートマシーンとクラスダイアグラム、さらに自動詳細化や自動証明、プロバビリティのサポートなども視野に入れる。
- 鉄道、自動車、宇宙。情報システム等の適用事例研究
  - ✓クルーズコントロール（Bosch）
  - ✓CBTC（STS）
  - ✓マイクロプロセッサ（ST Micro）
  - ✓Enterprise Service Communication（SAP）

形式手法に関連する欧州のランドマーク研究開発プロジェクト(4/5)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

・ProBツールは、英国EPSRCが助成するプロジェクトである「ABCD」と「iMOC」の成果を元に開発された。

ABCD (ESPRC(注))

- **名称**
  - コンポーネントベース設計によるビジネスに不可欠なシステムの自動妥当性検証（Automated Validation of Business Critical Systems with Component Based Designs）
- **期間**
  - 2000年2月～2003年9月
- **予算**
  - 424,893（ポンド）
- **参加組織**
  - Southampton大学（英）
  - IBM研究所（英）、International Computers（ICL）（英）、Roke Manor Research（英）
- **研究内容**
  - ビジネスに不可欠なシステムにおける形式モデリングの採用を促進するために、導入コストを下げ、形式モデリングを利用するメリットを高める。
  - システムとコンポーネントの汎用モデルのリポジトリを構築することでコストを下げる。
  - クリチカルシステムの検証や妥当性検討を実システムアーキテクチャで利用可能にすることで形式モデリングのメリットを高める。
  - 自動証明ツールをサポートすることで、コストを下げメリットを高める。
  - システム定義とアーキテクチャ設計にフォーカスを置くことで、システムインテグレータによるシステムのモデル化と提案するシステムアーキテクチャの妥当性検証を容易にする。

iMOC(ESPRC)

- **名称**
  - 抽象解釈とモデル検査を使う無限状態モデル検査（Infinite State Model Checking Using Abstract Interpretation and Model Checking）
- **期間**
  - 2000年2月～2002年6月
- **予算**
  - N/A
- **参加組織**
  - Southampton大学（英）
- **研究内容**
  - 自動で抽象モデルを生成する可能性の研究をする。
  - 現状の技術と部分評価自動制御と抽象解釈を組合わせ、自動的に無限モデル検査の抽象モデルを得る。
  - 上記は、ECCE（Partial Deduction System）とLOGEN（Offline Partial Evaluation System）ツールにより達成した。
  - プロジェクトは（ECCEに基づく）部分評価と抽象解釈システムの実装と理論研究からなる。
  - 実践的な取組みが、実事例で評価された。実事例のいくつかは、ABCDプロジェクトの成果を利用する。
  - 検証メソドロジ、メソドロジの可能性、能力の理論的研究、プロセス代数向けの無限状態モデル検査、経験的評価、より高位形式のモデリング（CSPやB）等

注:Engineering and Physical Sciences Research Councilの略、英国の物理科学や工学の研究および教育を支援する助成機関

形式手法に関連する欧州のランドマーク研究開発プロジェクト(5/5)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- 「Verifying Compiler」の開発を目的とした、GC6のプロジェクトが形式手法で検証されたソフトウェアのリポジトリを構築している。すでに実使用されているソフトウェアの再検証するテーマが多い。

Dependability Systems Evolution (GC6)

■ **期間**
- 2004年～

■ **背景**
- Grand Challenge (注) プログラムに設定された9つのテーマの中の一つ
- 10～15年の長期プロジェクト
- York大学 Jim Woodcock教授がチェアを務める。

■ **研究内容**
- 「Verified Software Repository」と「Verifying Compiler」（統合ツールスィート）を開発する。
- 「Verified Software Initiative」の活動により「Verified Software Repository」を構築する。
- システムの寿命全般にわたり高信頼性を維持するコンピュータシステムを可能にする構想
- この構想を実現するための技術の1つが「Verifying Compiler」であり、プログラムを実行する前に、その正しさを自動的に証明するツールである。
- このために2つの中心的規範を定める。
  - ✓ 理論はツールで実現されるべきこと
  - ✓ ツールは実システムで実証されるべきこと

Verified Software Repository(GC6の活動)

■ **期間**
- 2004年～

■ **背景**
- 検証ツールの利用を推進し、専門性の閾値を下げることで開発への取組みを促す。これにより、「Verifying Compiler」開発に向けた基盤を構築する。

■ **研究内容**
- MONDEX Software：10年前に開発されたスマートカードのシステムを複数の手法とツールで再検証
- Posix Filestore (York大学他) ：フラッシュメモリに適したPosixファイルストアインタフェースのサブセットの検証
- Operating system kernels：OSカーネルの検証
- Pacemaker（Boston Scientific）：VDM++を利用した心臓ペースメーカの開発
- FreeRTOS（Wittenstein high integrity systems）：形式手法未利用のリアルタイムOSソフトウェアを形式手法を利用して検証
- Tokeneer（Praxis）：バイオメトリクス認証システムの再検証
- Radio spectrum auctions：無線周波数割当競売プロセスの検証
- Hypervisors (Microsoft)：VCC／Verified C Compilerによる仮想化ソフトウェアのプログラム（Cとアセンブラ）の静的検証

■ **これまでの成果（例）**
- 複数の手法による再検証の結果、どの手法においても結果に大きな差異（開発時間、証明ステップ、自動化の程度発見された障害数）がないことが分かった（Mondex Case study）。

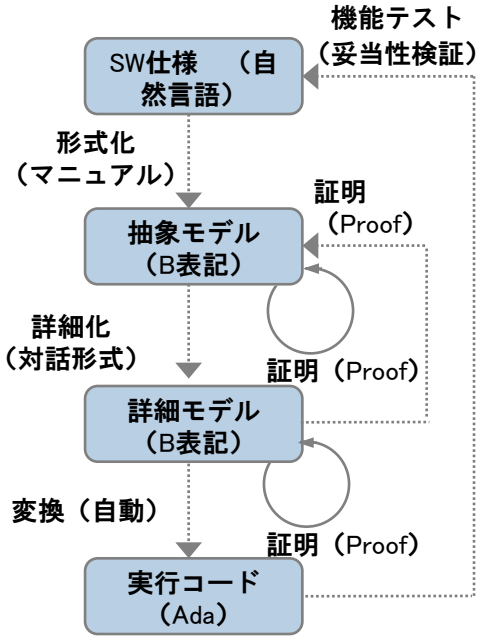
注：英国EPSRC(Engineering and Physical Sciences Research Council)の支援を受けUK CRC(Computing Research Committee)の助成により実施されている。形式手法の祖とも言われるTony Hoare の提唱により開始されたコンピュータサイエンスの学術研究プログラム(2004年から10～15年間)。GC1 からGC9のテーマが設定されている。研究成果は参加組織によりシェアされる。

形式手法開発プロセスの例：Bメソッド

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

・Bメソッドの開発プロセスをSTS社(Siemens Transport System)によるニューヨーク地下鉄のシステムへの事例で説明する。

形式化設計プロセス



▽ソフトウェア仕様

テキスト書類で記述され、機能分析の規範(SADT：注1)に沿って構成される。

■形式化

ソフトウェア仕様の内容を抽象モデルとして表現する。

▽抽象モデル

獲得された要求仕様が表現されたモデル

■詳細化(対話形式)

抽象モデル(抽象マシーンとリファインメント)から実装モデル(インプリメンテーション)を生成する。

▽具体モデル

抽象モデルと実装モデルからなる。

■証明

各コンポーネントに一貫性があり、詳細化される抽象モデルとの矛盾がないことを実証する。

■変換(自動)

具体モデル(B表記)を従来のプログラム言語に変換する。

▽実行コード

従来のプログラム言語(Ada、C++等)

■妥当性検証

実行される機能が要求仕様を満たしていることを検証する。

注1：SADT(Structured Analysis and Design Technique)：機能の階層でシステムを記述するソフトウェアエンジニアリングの手法

概要

対象

ニューヨーク地下鉄の最新化

列車制御システムの最新化　無線による制御システム(CBTC)

規模

抽象モデル：125,000行(B)

詳細仕様(手作業)：38,000行(B)

詳細仕様(自動)：110,000行(B)

証明責務

抽象モデル：38,500

詳細モデル(手作業)：19,000

詳細モデル(自動)：25,000

工数

不明

使用ツール

Atelier B (ClearSy)

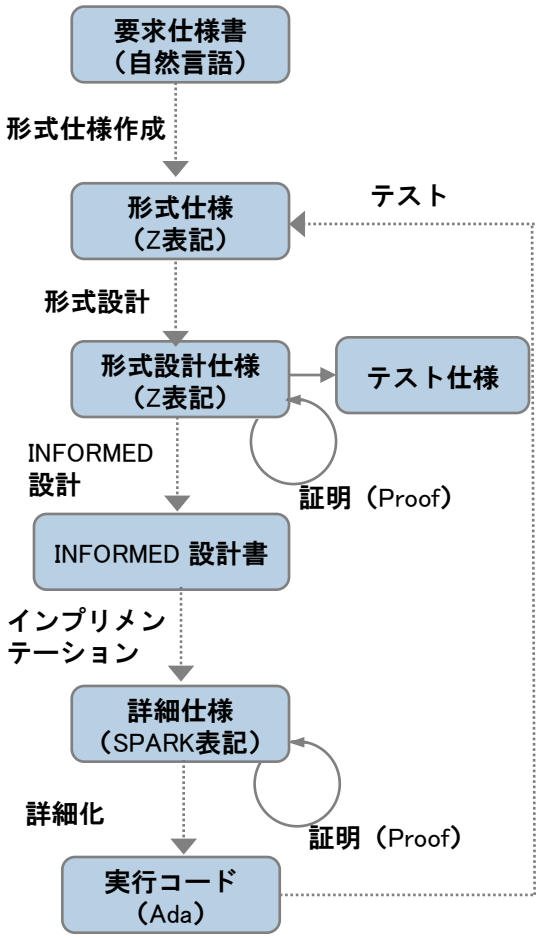
Edith B(STS内製)

形式手法開発プロセスの例：SPARKとZ表記

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- SPARKとZ表記による開発プロセスをPraxis社による「Tokeneer ID Station (TIS) 」の事例で説明する。

形式化設計プロセス



▽要求仕様書

■形式仕様作成

要求仕様書の内容をZ表記で形式化し記述する。

▽形式仕様

獲得された要求仕様を形式表現した機能仕様書

■形式設計

形式仕様を詳細化した形式設計仕様を生成する。

▽形式設計仕様

システムの詳細を定義する設計仕様

■INFORMED 設計

SPARKパッケージを生成するガイド等を定義する

▽INFORMED設計書

■インプリメンテーション

INFORMED設計書に沿って形式設計仕様書をSPARK表記（Adaのサブセットでコードとアノテーションからなる）

▽詳細仕様

SPARKで記述されたコードとアノテーション

■証明

SPARKのアノテーションとコードを読取り、アノテーションに定義された属性でコンパイルできることを自動で証明する。自動的に証明されない場合は、証明責務となり、マニュアルによる検査、あるいは「ヒューマンガイド定理証明機能「human-guided theorem Prover」で証明される。

■詳細化

プログラム言語(Ada)まで詳細化する。

概要

対象

「Tokeneer ID Station」(バイオメトリクスID認証ツールのアクセス管理セキュリティソフトウェア)

規模

形式仕様：100ページ(Z表記)

1.TIS コア

抽象モデル：16,564 行(SPARK)

実行コード：9,939行(Ada)

検証条件の証明：2,453

2.サポートソフトウェア

抽象モデル：2,240行(SPARK)

実行コード：3,697行(Ada)

工数

260人日(3名、12ヶ月体制)

使用ツール

REVEAL(要求分析)(Praxis)

SPARK Tool(Praxis)

fuzz type checker (Zのチェック)(OS)

AdaCore(Ada)

開発コスト

250,000ドル

開発者のスキル

レベルB ：1名(CS専攻、Spark経験、Z経験なし)

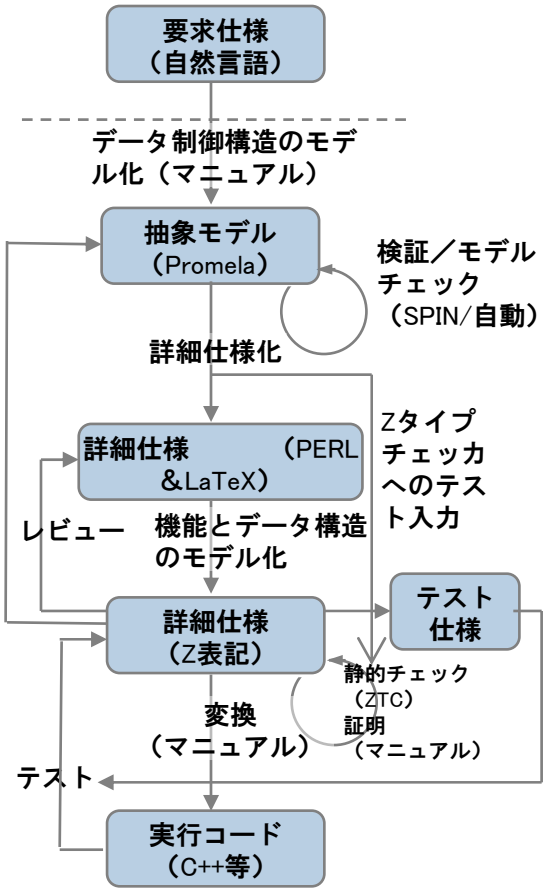
レベルC：2名(数学/CS専攻、Spark/Z経験なし)

形式手法開発プロセスの例：Promela/SPINとZ表記

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- Promela/SPINとZ表記による開発プロセスをCMG社によるオランダ防潮提開閉意志決定システムの事例で説明する。

形式化設計プロセス



▽要求仕様書、クライアントであるRWS(国家水路、水事業保全省)が記述した(700ページ)

■自然言語の要求仕様からデータ制御構造(外部システムとの通信モデルと制御モデル)をモデル化(Promela)

▽データ制御構造のモデル(抽象モデル)

■SPINによるモデル検証(ライブロックやデッドロックがないことを検証)

■Ward & Mellor(注1)手法を使いCASEツール(SDW)(注2)により詳細仕様を作成

▽PERL(注3)スクリプトとLaTeX(注4)で生成された詳細仕様書

■機能とデータ構造のモデル(Z表記)を抽出する。

▽機能とデータ構造のモデル(Z表記)

■静的チェック(文法、タイプ、完全性のチェック)をZTCツール(注5)により行う。

■Z表記を実行コードへ変換する(マニュアル)

▽実行コード

■テスト

注1：構造化分析/設計手法(SA/SD)、リアルタイムシステム向け、データフローダイアグラムに基づく構造化の手法
注2：Cap Volmac (Cap Gemini Sogeti group)が開発したケースツール
注5：Z表記の静的チェックツール
注4：インタプリタ方式のプログラミング言語とその処理系
注6：テキストベースの組版処理システム

概要

対象

オランダロッテルダム防潮提開閉意志決定システム

規模

要求仕様(機能仕様所)：700ページ

詳細仕様(Z表記)：29プログラム/20KLOC

実行コード

運用システムC++：200KLOC

シミュレータ/テストシステムC++：250KLOC

工数

2年(12名体制、20人年)

使用ツール

SPIN(モデル検証)(OS)

SDW(CASE)(Cap Volmac)

ZTC(静的チェック)(DEPAUL大学)

開発者のスキル

コアメンバ：5名(レベルA、Z読書き)

ソフトウェアエンジニア：7名(B、C Z読解)

専門家：3名(博士)

開発コスト

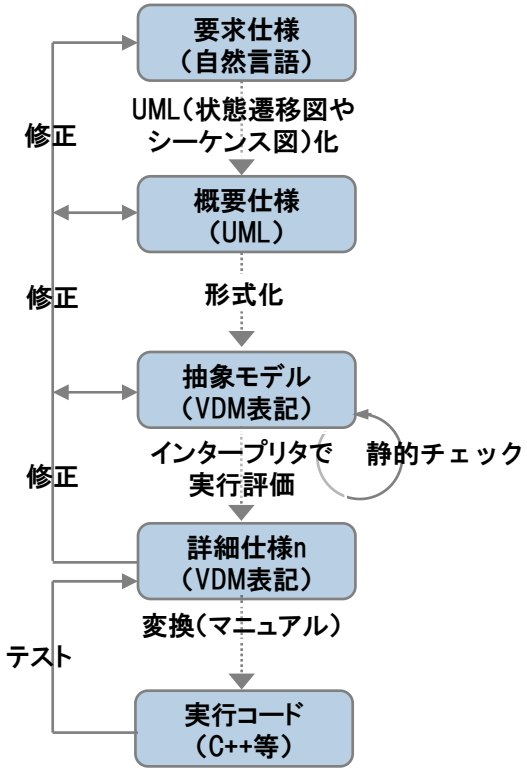
2.5～3百万ユーロ(ソフトウェア開発)(推定)

形式手法開発プロセスの例：VDM

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- VDMを利用による開発プロセスをフェリカネットワークスによるFelicaチップの事例で説明する。

形式化設計プロセス



▽要求仕様書（商品企画等からの要件、自然言語）

■要求仕様をUML表記に変換

▽UML図（状態遷移図やシーケンス図など）

■要求仕様やUML図から抽象モデル（VDM）の作成

▽コマンドやセキュリティ仕様

▽ファイルシステム仕様（データ構造）

▽無線／有線インタフェース仕様

▽フレームワーク仕様（データ構造とインタフェースに基づくテスト仕様）

▽Felica技術の基礎となるコマンド仕様

▽ファイルシステムとコマンド仕様に基づくセキュリティ仕様

合計100,000 LOC

■インタプリタで実行評価しながら詳細化を行う。

■実行コードへ変換（VDM Toolには自動コード生成機能があるがFelicaチップの事例では利用しなかった）。

▽実行コード（C++）110,000 LOC

概要

対象

Falicaチップ（スマートカードICチップ）

規模

プロトコルマニュアル：383頁

要求仕様（外部仕様）：677頁

形式手法記述：100,000LOC

（内テストケース60,000）

実行コード（C++）：110,000 LOC

工数：

3年3ヶ月、50～60名

仕様作成チーム：5～20人

ファームウェア実装チーム：15-20人

テストチーム：25～35人

使用ツール

VDM Tool

開発者のスキル

プロジェクト開始時には形式手法の経験者はいなかった

ソフトウェア開発の経験がないメンバもいた

情報技術の基礎知識はあるが、さまざまな経験年数のメンバが混在した

プロジェクト開始時、数名のメンバが5日間の形式手法トレーニングを受けた

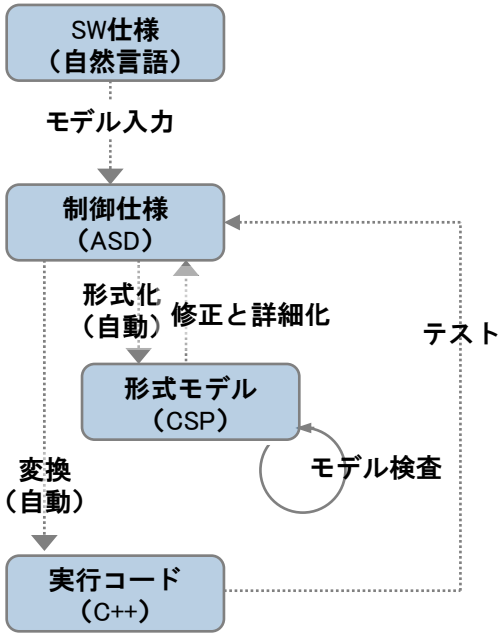
その他のメンバは社内のトレーニングを受けた

形式手法開発プロセスの例：ASD(CSP)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- ASD(CSP)を利用による開発プロセスをPhilips社によるCTスキャン開発の事例で説明する。

形式化設計プロセス



▽要求仕様書

■タブラー形式での画面入力によりASD（Analytical Software Design）制御モデルを作成

▽ASDモデル

■ASDモデルをCSP表記の形式モデルへ変換

▽形式モデル（CSP）

■形式モデルをモデル検査（FDRを利用）

■モデル検査結果に基づきASD表記を修正+詳細化

▽ASD表記

■形式モデルへの変換

■モデル検査を行いエラーがないことを確認

■ASD制御モデルを実行コードに変換

▽実行コード（C++、C#）

2ヶ月間の実結果データ

	LOCs	Users	Man days	ModelChecker Defects
C++	7697	1.5	10	36
C#	19684	5	69	387

2ヶ月間の実結果データからの推定

	ELOCs	Users	Effort (Man Hours)	Duration (Months)	Defects	ASD Cost	Effort Cost	Total Cost
C++	7697	1.5	320	8	0	€91,049	€24,000	**€115,049**
C#	19684	5.0	1656	6	0	€301,915	€124,200	**€426,115**

概要

対象

CTスキャン

規模

実行コード(全体)：27381 LOC

実行コード(C++)：7,697行

実行コード (C#)： 19,684行

工数

1,976 人時間(8ヶ月、6.5名体制)(2ヶ月間の実結果からの推定)

開発ツール

Verum ASD suite

ASD Model Builder

ASD Model Checker

ASD Code Generator

開発費用

541,164ユーロ(SaaS方式でツール利用)(2ヶ月間の実結果からの推定)

形式手法を使用しない開発費用の推定　848,811ユーロに比べ36%のコスト削減

開発チームのスキル

1.5名：C++の開発経験者(形式手法の経験なし)

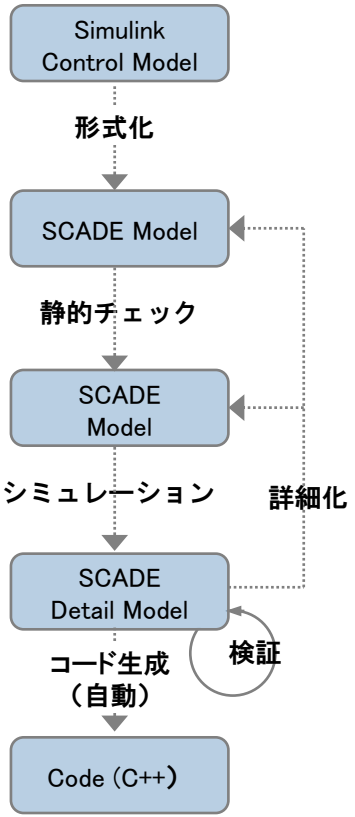
5名：C#の開発経験者(形式手法の経験なし)

形式手法開発プロセスの例：SCADE

SEC
Software Engineering
for Mo・No・Zu・Ku・Ri

・SCADEを利用による開発プロセスをAirBus社の開発事例で説明する。

形式化設計プロセス



▽Simulinkの制御モデル

■Simulinkのモデルを入力とし、Scadeのモデルに変換し（SCADE-Simulink Gateway）、エディタにキャプチャーする（SCADE Editor）。

▽SCADEモデル

■SCADEモデルの静的チェック（Syntax Checking、Semantic Checking）を行い（SCADE Checker）、結果を元にScadeモデルを修正する（SCADE Editor）。

▽SCADEモデル

■モデルのシーケンシャルなシナリオをインタラクティブにシミュレーションする（SCADE Simulator）。

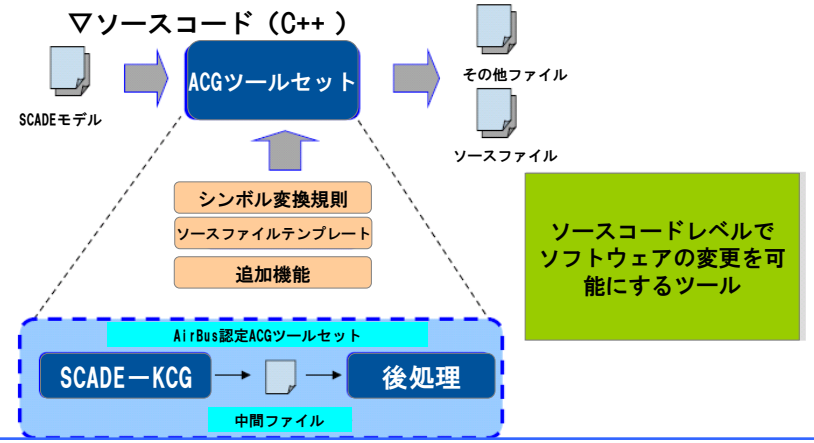
▽SCADEモデル

■SCADEモデルを検証する（Prover Plug-In enables 検証）

▽SCADEモデル

■AiyBus ACGツールセットでSCADEモデルをソースコードへ変換する。

▽ソースコード（C++）



概要

対象

Airbus A340（A500/600、A380）
Flight Control System
Fly-by-Wire-Controls
Display Computer
Warning & Main Computer

規模

785,000LOC（A340 Flight Control System）

自動コード生成の適用比率

各システムのLOCのうち自動コード生成されたLOCの割合

70%:Flight Control System
70%:Fly-by-Wire-Controls
50%:Display Computer
40%:Warning & Main Computer

開発ツール

SCADE-Simulink Gateway

AirBus ACG ツールセット
（SCADE-KCGを含む）

ーターゲットハードウェアに応じて、生成されたソースコードをモデルに影響を与えずにカスタマイズが可能

Prover Plug-In enables verificationツール
（A380 flight control systemで採用）

形式手法開発プロセスの例：開発工程の工数分布イメージ

SEC
Software Engineering
for Mo･No･Zu･Ku･Ri

・各手法による典型的な開発プロセスのおよその工数比率を推定した。

形式手法開発チームのスキル(1/2)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- 主要なプロジェクトの開発チームメンバのスキルを示す。

対象システム（開発手法）	所属	人数	レベル（経験／知識）／人数	役割	備考
鉄道軌道システム（B）	Alsthom（メーカ）	2名	A（信号システムの専門家）／1名 B（システム技術者）／1名	仕様作成 アーキテクチャ開発 妥当性検証	Alsthomのプロジェクトの場合は、サプライヤ（ClearSyやSyaterel）が主体となるプロジェクトが多く、STSのプロジェクトの場合は、STSが主体となりサプライヤがSTSのリソースを補う体制が多い
	Systerel（サプライヤ）	6名	A（12年のB手法による開発経験、技術系大学卒業）／1名 B（7年のB手法プロジェクト経験）／1名 C（3年未満の経験）／4名	仕様作成 実装 インテグレーション	
パリ地下鉄自動制御システム（B）	RATP（ユーザ）	15名	シニアとジュニアのSWエンジニア：15名、Bを記述するスキルは必要ない。妥当性検証のために読込むスキルが必要	仕様作成、妥当性検証	仕様作成（モデル化）、モデル詳細化にはBの記述能力だが必要だが、実行コードへのコーディング、テスト、妥当性検証では読解ができればよい。 左記事例は、B手法採用の最初のプロジェクトであったため、工数が現在の数倍かかった。現在は、RATP：1～2名、STS：5名程度のプロジェクト規模となっている。
	STS（メーカ）	35名	A（10年以上の経験者、PHD1名、SW技術者1名）／3名 BまたはC（SW技術者として入社し、社内研修を経て、形式手法のスキル（Bの記述と読解）を習得）／32名	プロジェクト管理、プロセス管理 モデリング、検証、コード生成、テスト、妥当性検証	
パリ地下鉄プラットフォームドアの制御（B）	ClearSy（サプライヤ）	4名	B（経験年数5年以下のSW技術者；PM）／1名 B（シニアSW技術者；安全技術者）／1名 C（ジュニアSW技術者；SW実装）／1名 C（ジュニアSW技術者；検証技術者）／1名	仕様作成 実装 テスト	
サンパウロ地下鉄プラットフォームドア制御（SCADE）	POSCO（サプライヤ）	3名	システム開発技術者（形式手法経験なし） ソフトウェア技術者（形式手法経験なし）	仕様作成 実装 インテグレーション	当初、B手法の採用を検討したが、3ヶ月という短い納期に間に合わせるため、導入が容易なSCADEを採用することとなった。

形式手法開発チームのスキル(2/2)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- 主要なプロジェクトの開発チームメンバのスキルを示す。

対象システム（開発手法）	所属	人数	レベル（経験／知識）／人数	役割	備考
防潮提開閉意志決定システム（Promela/SPIN）	CMG	12名	A（プロジェクト経験10年前後、Zの記述、SPINのモデリングができる）／5名 A（形式手法専門家、博士）／1名 BまたはC（プロジェクト経験5年以下、Zの読解ができる）／7名	仕様作成 アーキテクチャ開発 実装 テスト	Z表記の学習に時間がかかった（6ヶ月）。現在、プロジェクトに係わった技術者はLOGICA社などで形式手法のプロジェクトを実施している。
	Twente大学	2名	A（形式手法専門家、博士）／3名	開発プロジェクトに専門的助言をする	
バイオメトリクスID認証ツールのアクセス管理セキュリティソフトウェア（SPARK）	Praxis	3名	A（Zの読書き、セキュリティ技術）／2名 B（Zの解読）／1名	仕様作成 アーキテクチャ開発 実装	■一般的なチーム構成 電子技術者、システム技術者、数学的素養がある技術者の組合わせ A：20%（実装技術だけでなく、特定分野の専門スキルや管理能力を有す） B：30%（Zの記述と読解ができる） C：50%（Zの読解できるが記述できない）

形式手法の成功事例：Bメソッド発展の経緯

SEC
Software Engineering
for Mo・No・Zu・Ku・Ri

・Bメソッドとそのツールが実用化された経緯を示す。

Jean-Raymond Abrial.

1993年

協力

Digilog社（注4）

Atelier B開発

GEC Alstom（注1）

買収

Steria社（注5）

技術協力

プロトタイプツールが、3百万ユーロ／5年のプロジェクトで強化された

Diginext社（注6）

技術者の移動

スピンアウト

INRETS（注7）

RATP（注8）

SNCF（注9）

2001年

技術者の移動

Atelier B 販売

ソフトウェア開発

Systerel社（注11）

ClearSy社（注10）

Alstom（注2）

STS（注3）

システム販売

NY地下鉄（STS）、サンパウロ地下鉄（Alstom）、北京地下鉄（Alstom）等へ海外展開

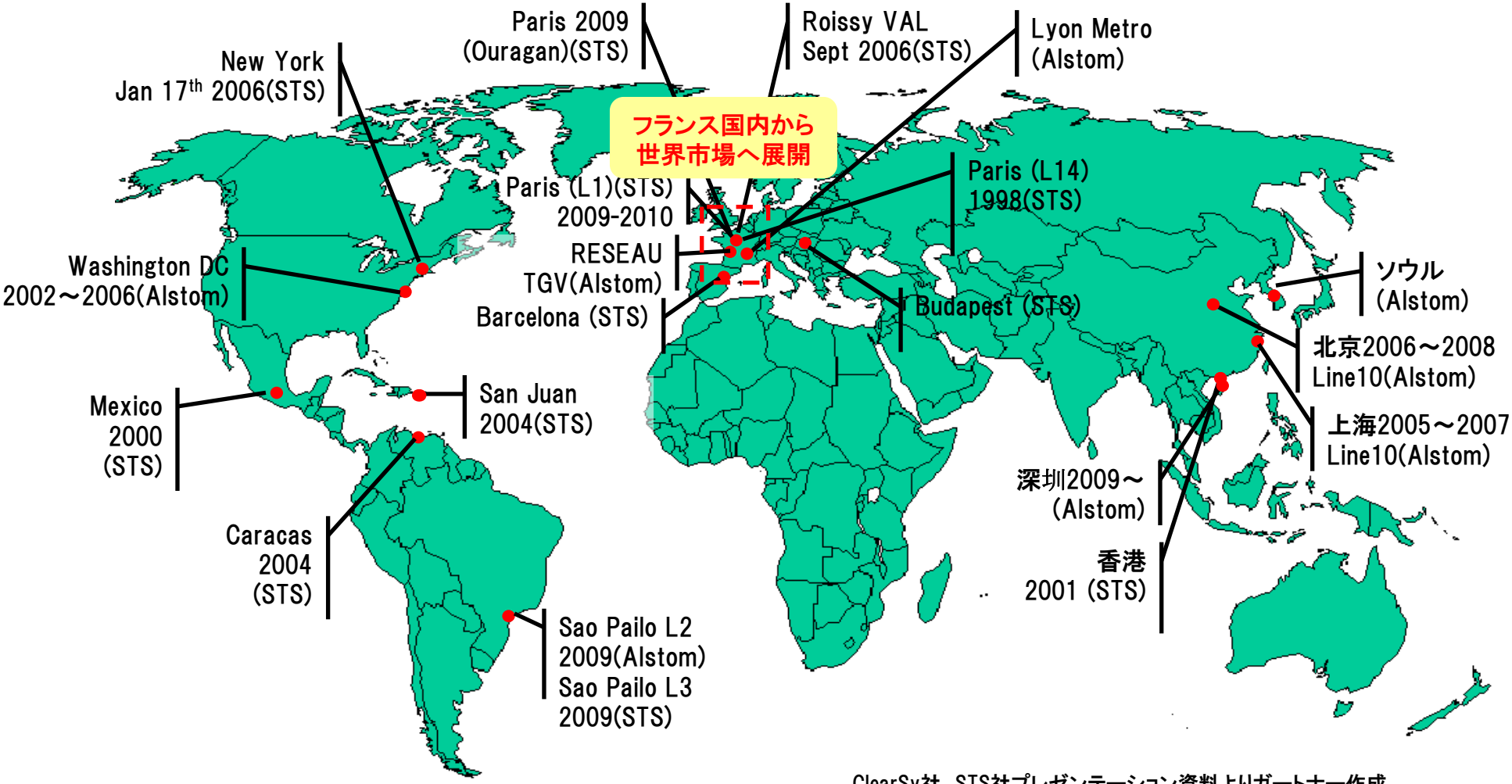
ソフトウェア開発

注1：現社名Alstom（フランスの重電・重工メーカ）、GECは英の企業で当時合弁、SACEM（自動速度制御システム）は、1982～1985年にRATPが主導するパリPER線向けプロジェクトで Jeumont-Schneider（後、GEC-Alstomが買収）、Matra（現STS）、CSEE（現ANSALDの傘下）3社の共同開発（Bメソッド採用）、後にAlstom、STSがそれぞれSACEMを継続して開発している。
注2：TGVや地下鉄などの鉄道設備、車両、発電設備を開発、販売するフランスの重電・重工メーカ
注3：95年に独Siemensと仏輸送機器メーカのMatraが合弁で設立、98年にSiemens100%傘下となり、01年に現社名Siemens Transportation Systems（STS）（仏）へ改名
注4：フランスのコンピュータサービス企業、1993年Steria社に買収された。
注5：フランスのコンピュータサービス（ソフトウェア＆システムインテグレーション）企業、1983年にDigilog社を買収した。
注6：ソフトウェアとコンピュータベースシステムの開発企業
注7：フランスの交通と安全に関する国立研究所、運輸省の研究機関
注8：パリ交通公団（RATP：Régie Autonome des Transports Parisiens）、フランスの首都パリとその周辺部の公共交通機関を運営する。
注9：フランスの国鉄（Société Nationale des Chemins de fer Français）
注10：Steria （仏総合ITベンダ）65%、Teamlog（同NWベンダ）35%出資のBメソッドの形式手法ツールAtelier B を開発するツール／SIベンダ
注11：Bメソッドを主とする組込み系のリアルタイム・システムやクリティカル・システムの開発や検証に強みを持つ小規模SIer

形式手法の成功事例：鉄道システムへのBメソッド適用(2/2)

SEC
Software Engineering
for Mo・No・Zu・Ku・Ri

- Alstom社とSTS社(Siemens Transport System, 旧Matra社)は、RATPやSNCFなどのフランス国内の鉄道サービス企業に留まらず、ニューヨーク、北京、サンパウロ地下鉄などの世界市場へAtelier Bを利用して開発されたシステムを展開している。
- STS社あるいはAlstom社で開発したシステムが世界の地下鉄の80%で採用されているという。



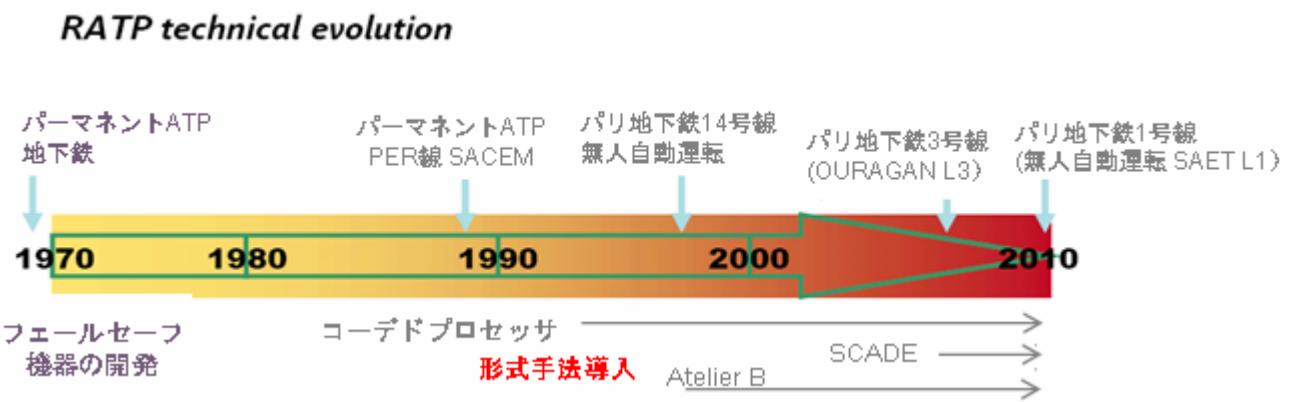
ClearSy社、STS社プレゼンテーション資料よりガートナー作成

形式手法の成功事例:RATP(パリ交通局)／適用事例

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- RATP(パリ交通局)では、ClearSy社と協業してBメソッドツール(Atelier B)の開発に係り、Bメソッドを採用して開発されたシステムをSTS社やAlstom社から導入してきた。
- しかしながら、RATPでは Bメソッドを採用して開発されたシステム以外に、SCADEを利用して開発されたシステムの導入を計画している。
- その理由として以下の2項目が指摘された。
  - 公共機関であるRATPが特定企業やツールの採用をすることなく、できる限り公平に優良な技術を採用する
  - Atelier Bを採用した開発では、ハードウェア開発のプロセスとソフトウェア開発のプロセスの協業が非効率な面があった。SCADEを利用することで、MATLAB/Simulinkとのインタフェースがとれるため、ハードウェアとソフトウェアの開発の協業プロセスの効率の改善が期待できるという。

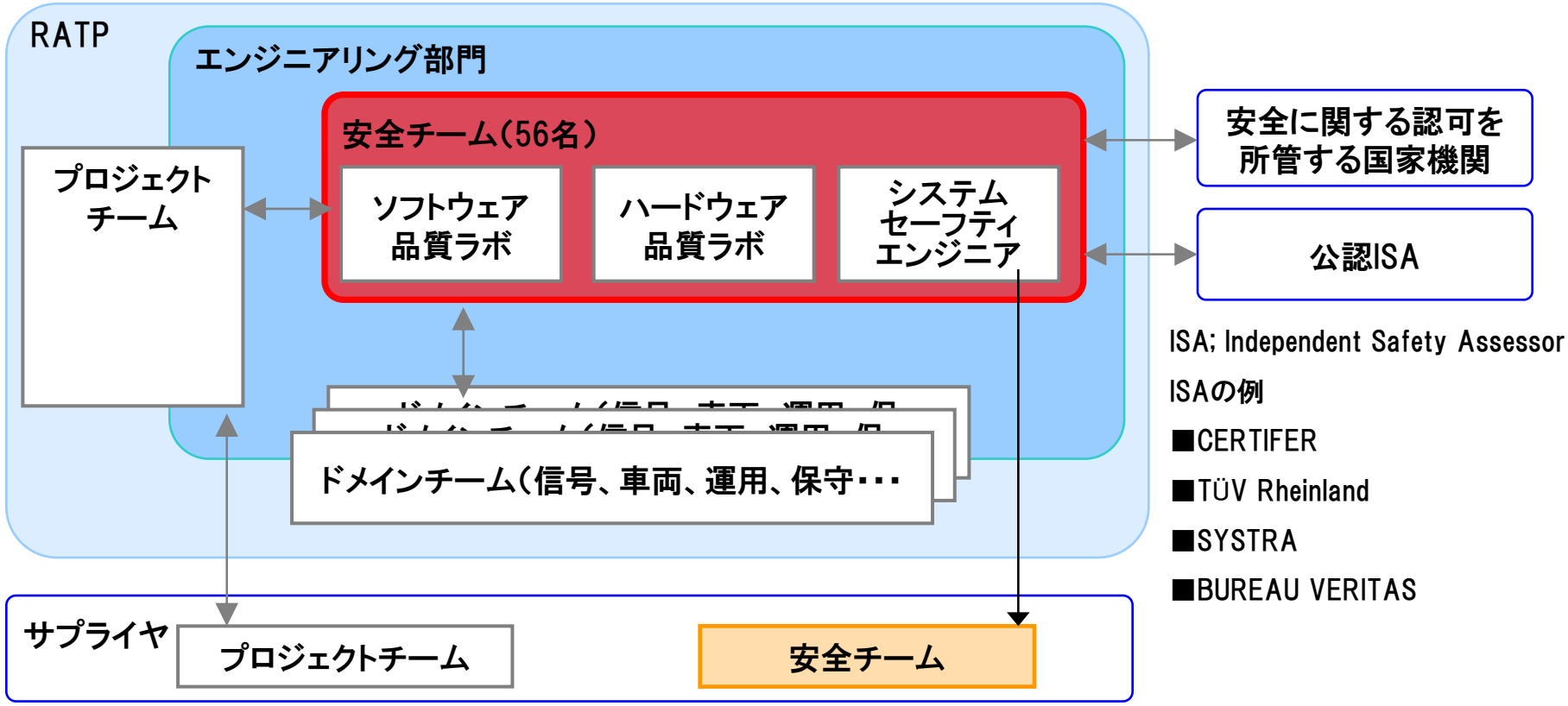
運用開始時期	プロジェクト名	対象機器	サプライヤ	ツール
2009年12月	OURAGAN L3	車載設備(On Board)	STS	Atelier B
		軌道設備(Way side)	ANSALDO	SCADE Proof Tool Kit
2010年	SAET L1	車載設備(On Board)	STS	Atelier B
		軌道設備(Way side)	STS	Atelier B
2011年	OURAGAN L13	車載設備(On Board)	Thales	SCADE
		軌道設備(Way side)	Thales	SCADE
2011年	OURAGAN L5	車載設備(On Board)	AREVA-TA	SCADE
		軌道設備(Way side)	STS	Atelier B
2009、2010年	PMI L1, L12	車載設備(On Board)	ー	ー
		軌道設備(Way side)	Thales	Proof Tool Kit



形式手法の成功事例:RATP(パリ交通局)／開発組織

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- RATP(パリ交通局)のエンジニアリング部門には、56名で構成される安全チームがあり、ソフトウェア品質ラボ、ハードウェア品質ラボ、システムセーフティエンジニアから構成される。このうちソフトウェア関連の技術者は30名。
- ソフトウェア技術者、ハードウェア技術者、システムセーフティ技術者がプロジェクトチームに加わり、サプライヤのプロジェクトチームと協調して、仕様の作成、妥当性検討、受け入れテスト、セーフティ分析、セーフティケースの作成を行う。ソフトウェア品質ラボのほとんどのソフトウェア技術者は、形式手法の基本概念を理解している。仕様作成、妥当性検討では、B表記を理解する必要(読める)はあるが、記述する必要はないため、ソフトウェア技術者の形式手法のスキルは、基本概念の理解とB表記の理解(読むことができる)にとどまる。

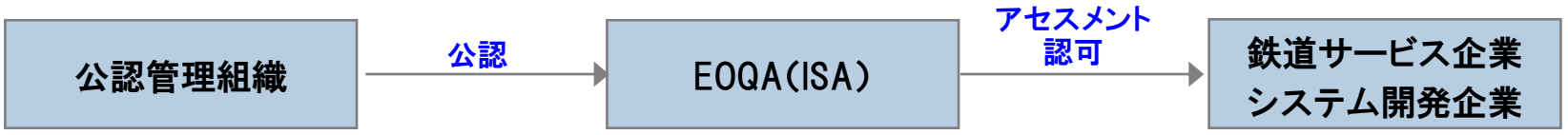


RATPプレゼンテーション資料よりガートナー作成

参考:フランスのアセスメント機関:ISA(Independent Safety Assessor)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- ISO/IECなどの国際標準の場合には、任意のISAが適合性を認証できる。
- CENELEC(欧州電気標準化委員会)では、システムとすべての部品がISAにより評価されることを義務付けている。
- ドイツでは、アセッサが同じ社内に存在する場合もあるが、フランスの場合は、ISAは社外の第三者機関であることが要求される。
- 認可が必要となるシステムのアセスメントは、公認管理機関が公認した公認団体であるEOQA(ISA)によるアセスメントが必要となる。
- 例えばETRM(欧州全体で使用可能な信号保安装置)は、公認機関のみが認定することができる。



フランスの公認管理組織
■EPSF
■STRM-tg

EOQAとなっているISAの例
■CERTIFER
■TÜV Rheinland
■SYSTRA
■BUREAU VERITAS

鉄道サービス企業の例
■RATP(パリ交通局)
■SNCF(フランス国鉄)

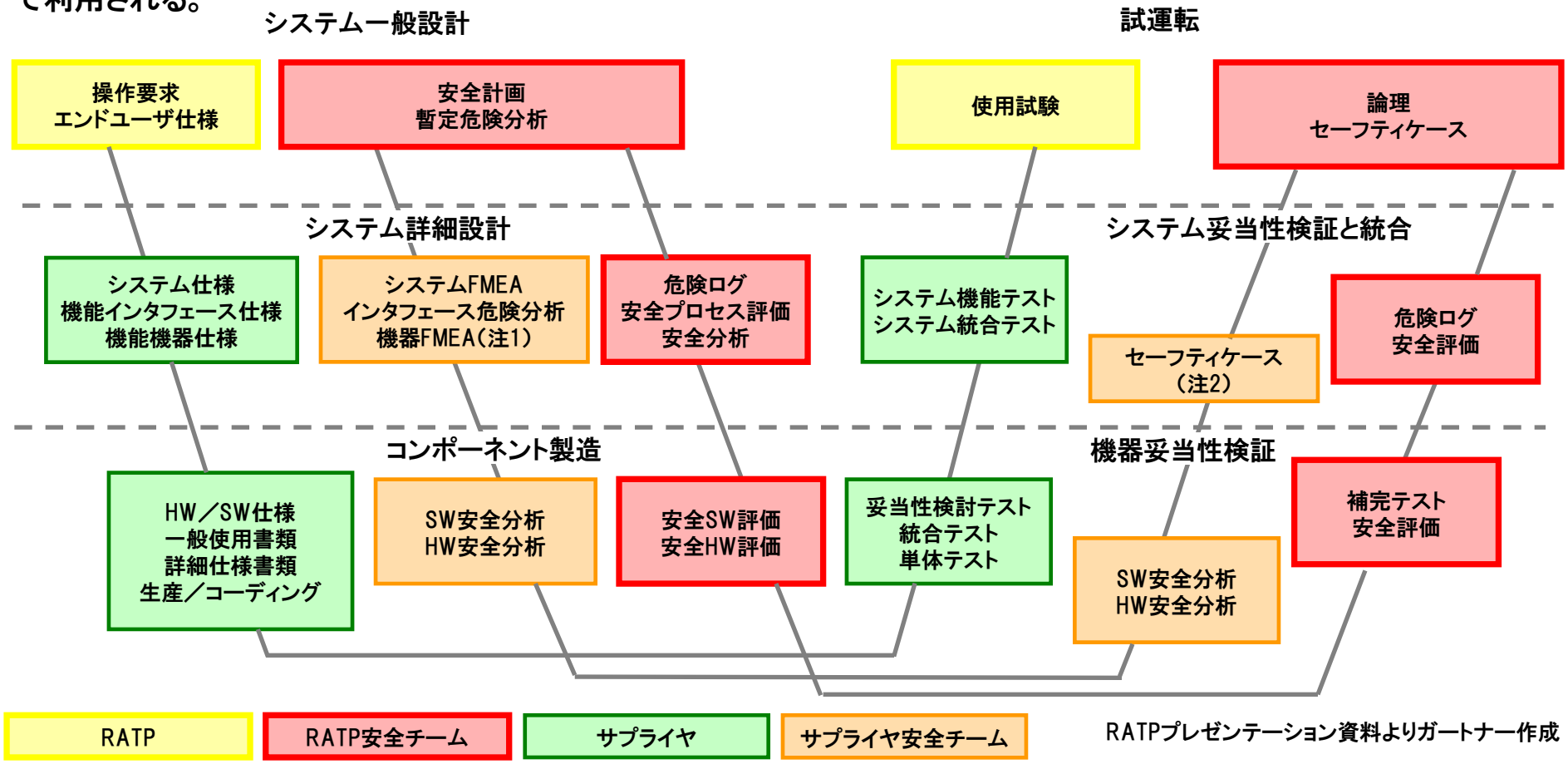
システム開発企業の例
■STS社(Siemens Transport System)
■Alstom社

- EOQA(Expert or Approved and Qualified Organization):ISO17025の認証を取得し、認可管理組織から認定を受けた公認ISA(NOBO)
- NOBO(Notified body):公認団体、運輸省により公認された団体でアセッサの任命ができる。
- ISA(Independent Safety Assessor):システムのアセスメントを実施する組織
- ISO/IEC17025:試験所及び校正機関が特定の試験又は校正を実施する能力があるとして認定を受ける場合の一般要求事項を規定したもの
- EPSF(French public railway safety authority):EOQAの公認をISAに与える認可管理組織(公共鉄道)
- STRM-tg (French technical department for lifts and guided transport systems):EOQAの公認をISAに与える認可管理組織(公共鉄道以外の軌道輸送システム)
- ETRM(European Rail Traffic Management System):EUが先導し開発する国境を越えても相互運用を可能にし、欧州全体で共通に使用できる信号保安装置
- CERTIFER(仏):鉄道専門の認証サービス機関
- TÜV Rheinland(独):製品安全試験・認証、マネジメントシステム認証を行う国際的第三者機関
- SYSTRA(仏):鉄道と公共輸送を専門にする技術・コンサルティング企業
- BUREAU VERITAS(仏):第三者検査・認証機関、ISO等国際規格や独自規格の認証サービス提供

形式手法の成功事例:RATP(パリ交通局)／開発プロセス

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- RATP(パリ交通局)におけるサプライヤと協調して行われる　開発／妥当性検討／安全分析 のプロセスを示す。
- 設計、製造、テスト、検証のV字開発プロセスにおいてRATPとサプライヤが協調しながら、安全システムの設計を進める。
- 形式手法ツール(SCADE、Atelier B等)は、機能仕様設計、コード自動生成、安全分析、セーフティケース作成などの工程で利用される。



注1:Failure Mode and Effect Analysis(故障モードとその影響の解析)は、故障・不具合の防止を目的とした、潜在的な故障・不具合の体系的な分析方法
注2:システムの安全を示す書類で、安全性を示す証拠書類(Evidence)と安全性の論拠を示す論拠(Argument)からなる。

形式手法の成功事例：STS／プロジェクトチーム

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- 安全系ソフトウェアの開発は、開発チーム、テストチーム、安全チームにより組織される。安全チームの役割は、ソフトウェアがシステムの安全性において妥協しないことを確認することにある。形式手法開発のチームの構成もこれに非常に似ていて、従来のチームに形式手法の専門家が加わる（形式手法支援チーム）ことで、方法やツールの支援をする。
- STSにおける典型的なアプリケーションの規模は、500ファイル、数10万行のBモデル、数万行の証明責務、10～15のサブモデルであり、プロジェクトチーム全体の規模は10名程度となる。

形式手法支援チーム	形式手法支援チームの氏名は、ベストプラクティスを収集し、普及することにある。1.「B開発ガイド」を保守する。この書類は、方法論のリポジトリであり、形式手法プロジェクトの経験を蓄積する。2.ツールと方法をサポートし、開発者に提供することで、技術レビューによりBモデルが証明可能で効率的であることを示す。3.重要な開発工程に参加し、指導する。 ほとんどの開発メンバはB表記の経験者であり、形式手法支援チームの役割は薄れつつある。
開発チーム	開発チームは、ソフトウェア仕様の記述からシステムへの統合までのすべてを実行する。1.ソフトウェアの仕様を記述する。2.抽象モデルの仕様を形式化する。3.抽象モデルをリファインし、従来のプログラム言語に変換可能にする。3.Bメソッドに要求されるすべての証明責任（Proof Obligation）を証明する。4.Adaコードのソフトウェアを生成する。コードは自動的に生成されるか、アプリケーションのローレベル入出力プリミティブで手作業で記述される。
テストチーム	ソフトウェアが合格すべきテストを定義、実行する。
安全チーム （妥当性検証チーム）	妥当性検証チームは、他のチームとは独立して、各開発チームの中間成果物を分析する。チームの目的は、開発の早い段階で致命的な欠陥を特定することにある。1.ソフトウェアの仕様の妥当性を検証する。2.ソフトウェア仕様の重要な]仕様すべてをチェックし、抽象モデルに採りこむ。3.アプリケーションのAdaコードをチェックし、形式的に開発されていない部分をチェックする。4.インタラクティブな証明（Proof）の妥当性を検証する。4.重要な仕様に関してのテストを定義し、実行する。これらの活動の結果発生した問題が報告され、開発チームにより修正される。

形式手法の利用を推奨する国際標準：主要なソフトウェア開発標準の対応状況

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- ソフトウェア開発に関連する国際標準の形式手法への対応を示す。

国際標準	対象	形式手法についての記述
ISO/IEC 61508	機能安全（一般）	SIL2、SIL3で推奨、SIL4で強く推奨（別頁参照）
CENELEC EN 50128	機能安全（鉄道）	SIL1、SIL2で推奨、SIL3、SIL4で強く推奨（別頁参照）
ISO/IEC 26262	機能安全（自動車）	ISO/WD26262（注）のASIL Cにおいて形式手法と仕様作成でのコンピュータツールの使用が要求される予定
RTCA DO-178C	航空	DO-178Bでは形式手法に触れている程度だが、DO-178CではSWの設計と仕様での形式手法の利用が明確な記述となる予定
UK Def. Stan. 00-55	防衛	形式手法を有効なメソッドとする記述がある。
ESA 91　PSS-05-0	宇宙	ZやVDMの使用を検討すべきと記述
ISO/IEC 15408	セキュリティ	EAL7でサブシステムレベルまで形式表現されることを評価基準とする（別頁参照）
IEC 60880	原子力	記述なし
EN 62304	医療機器	記述なし
EN 50271·EN 50402	ガス測定、検出装置	記述なし
FIPS 140-2	米国政府の連邦情報処理標準	レベル4で形式手法について言及

注：2010年制定予定のワーキングドラフト

国際標準・調達規定における形式手法の適用

SEC
Software Engineering
for Mo・No・Zu・Ku・Ri

- 政府・公共調達等における形式手法適用状況

- WTO政府調達協定：GPA（The Agreement on Government Procurement）
  - WTOの規定では政府が購入する製品やサービスは国際標準や国内の技術規制に従い調達することを規定
  - 形式手法適用の製品やサービスの購入につて直接規定するものではない
  - しかし、「国際標準に従う」ことは、先に述べた ISO/IEC 61508などの形式手法の採用を記載する標準に沿った政府調達をすることを規定していると言える。

- 英国軍事調達：Def Stan 00-55
  - Def Stan 00-55（REQUIREMENTS FOR SAFETY RELATED SOFTWARE IN DEFENCE EQUIPMENT）では、英国の軍事調達において安全系ソフトウェアを購入する場合の形式手法の適用について規定している。

- 米国国家情報保証調達ポリシー(National Information Assurance Acquisition Policy) (2000 年1月発行)
  - 国防総省が議長を務める国家安全保障通信・情報システムセキュリティ委員会（NSTISSC）、現在の国家セキュリティシステム委員会（CNSS）、によって策定された、情報保証（Information Assurance：IA）とIA対応のIT 製品の調達に関する方針の規定

形式手法の教育／トレーニング(1/2)

SEC
Software Engineering
for Mo･No･Zu･Ku･Ri

ClearSy(仏)

■ 実施状況

- Atelier-Bをトレーニングを含めて提供
- ClearSy Academy設立： ブラジル、スイス、独、カナダ、フィンランド等
- 4レベルのコース

■ コース概要

- Method B Training Level 1 : Understanding B
  Bメソッドの規範の基礎の理解
  4日間、2,130ユーロ
- Method B Training Level 2 : Practicing B
  Bプロジェクト開発規範の理解
  「good」Bモデル構築の練習
  B言語のコンセプト：上級.
  イベントドリブンBについて
  4日間、2,130ユーロ
- Method B Training Level 3: Prove B
  Bモデルの証明を学習するワークショップ
  3日間、1,780ユーロ
- BART Tool Training Level 4：自動詳細化
  BARTツールによるBモデルの自動リファイン（詳細化）を学習する。
  3日間、1, 780ユーロ

Praxis HIS(英)

■ 実施状況

- Praxisでは1990年からSPSRKのトレーニングコースを実施している。年間約10回のコースが開催され、60名程度が参加する。現在までに500名以上の参加者があったと推定される。
- SPARK以外のZなどの外部向けコースは実施していない。社内技術者向けにZのトレーニングを行う。
- トレーニングはPraxis社本社（Bath UK）で行われる。

■ コース概要

- Course 1: Two Day Overview
  SPARKを利用した高信頼性ソフトウェア開発技術の規範と実践に関する管理者と技術者向けチュートリアル
- Course 2: Software Engineering with SPARK
  高信頼性ソフトウェア開発の規範と認証の要求について示し、SPARK Examinerの説明と実習を管理者、規制担当者、技術者向けに行う。
- Course 3: Advanced SPARK Program Design and Verification
  SPARKを利用した証明や検証について習得する上級コース
- Course 4: Concurrent Software Design with RavenSPARK
  Ada95のサブセットを定義するRavenscarプロファイルについて説明する。
- Course 5: Introduction to the Proof Checker
  SPARK 証明チェッカについて説明する。
- Course 6: UML to SPARK
  UMLからSPARKを生成する理論について説明する。

形式手法の教育／トレーニング(2/2)

SEC
Software Engineering
for Mo･No･Zu･Ku･Ri

Esterel(仏)

■ 実施状況

- トレーニングは Elancourt (Paris)、Villeneuve-Loubet (Riviera), Toulouseのトレーニングセンタ、あるいは顧客サイトで行われる。
- SCADE開発プロセスに特化した5つのモジュールを提供する。各モジュールは3ないし4のコースで構成される。

■ コース概要

- Module M1: Model-based design with SCADE
  SCADEでソフトウェアに割当てられるシステム要求の獲得とトレーサビリティについてのコース（4コース）
- Module M2: Model-based verification
  SCADEソフトウェア要求がソフトウェアに割当てられたシステム要求に従うことを検証する方法についてのコース（3コース）
- Module M3: Model-based automatic code generation
  SCADE KCGでコード生成し、最終アプリケーションに統合する方法についてのコース（3コース）
- Module M4: Project Management with SCADE
  SCADEプロジェクトを管理するためのSCADEメソドロジと機能についてのコース（4コース）
- Module M5: SCADE Advanced
  SCADE開発メソドロジについての上級者コース（4コース）

Verum(蘭)

■ 実施状況

- 2010年からStarting ASDとMastering ASDの各コースを半年に4回程度の頻度で実施する。参加者は、同社のサービスを購入し、ASDスィートを利用するユーザ。

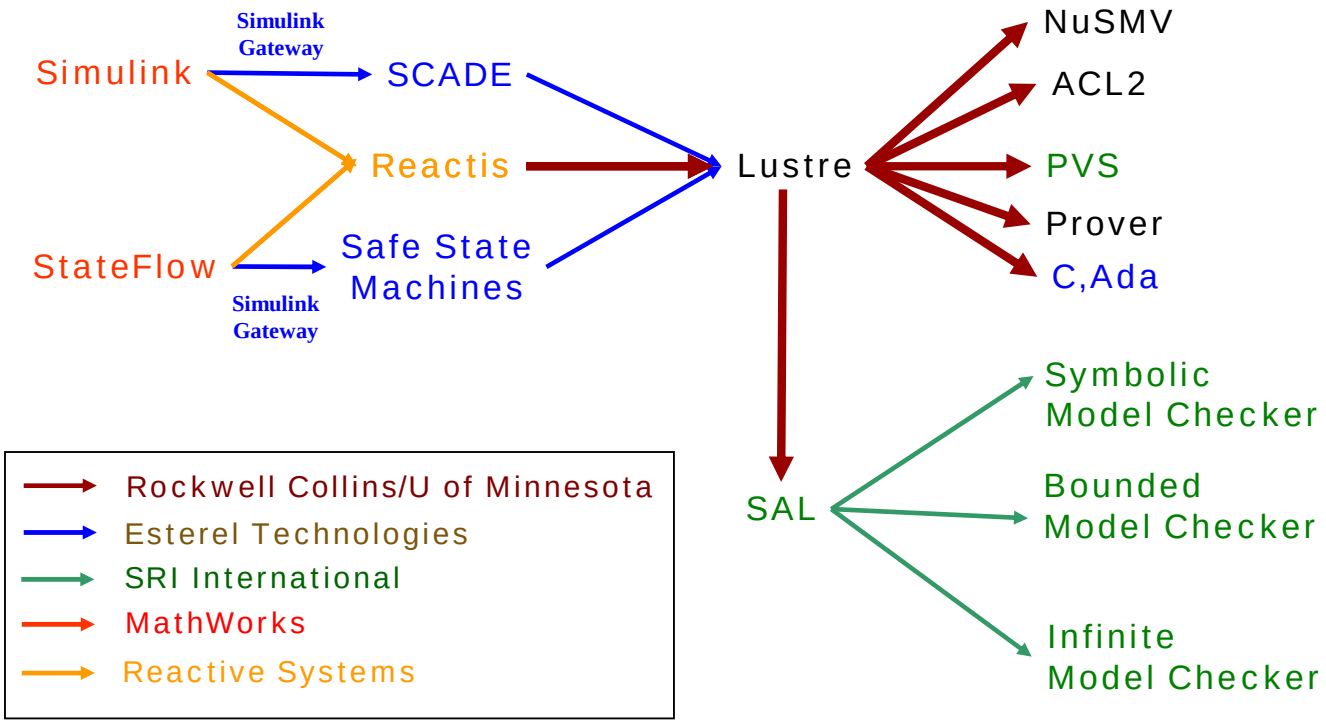
■ コース概要

- Starting ASD
  3日間コース：1,800ユーロ
  ASDの基礎を学習する3日間のコース。ASDインタフェースと設計モデルの構築と検証の方法、自動コード生成など、ASDスィートの使い方を学習する。
  このコースは、ASDの利用を始めるソフトウェアアーキテクト、ソフトウェア設計者を対象とする。ASDスィートについての事前知識は必要としない。
- Mastering ASD
  2日間コース：1,200ユーロ
  前コースの拡張コース。複数のアーキテクチャ／設計パターンと効果的なASDの利用方法を学習する。
  また、いくつかの先進的モデルチェックとコード生成の機能についても学習する。
  このコースは、ASDを個々のソフトウェアコンポーネントではなく、システムに適用するソフトウェアアーキテクトと設計者を対象とする。

米国における形式手法の適用事例:Rockwell Collins(1/2)

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- Rockwell Collins社おける形式手法の事例
  - NASAとRockwell Collinsがファンドする航空安全プログラム(2001～2006)において、最新航空システムへの形式手法の実応用プロジェクト(Methods and Tools for Flight Critical Systems Project)が実施された。
  - このプロジェクトは、Rockwell Collinsと Minnesota 大学が共同で実施したもので、よく使われる複数のツールモデル言語とモデルチェッカ、定理証明ツールを接続するトランスレータ(下図→部分)を開発した。下図にその接続の全体像を示す。



Reactics:Reactive Systemsが開発、販売するモデルベースの自動テスト生成、プロパティ検証ツール。

NuSMVは:ITC-IRSTの自動推論部門の形式手法グループ、Carnegie Mellon大学のモデル検証グループ他で共同開発されたシンボリックモデル検査器

ACL2 (A Computational Logic for Applicative Common Lisp):ソフトウェアのシステムで、プログラム言語、ファーストオーダロジックの拡張可能理論と機械的定理証明器からなる。帰納論理理論の自動推論をサポートし、主にソフトウェアとハードウェアの検証に使われる。

PVS:検証システム。サポートツールと定理証明器を統合する仕様言語である。SRI Internationalが開発しているプロトタイプ。

Prover:スウェーデンのProver Technolpgyが開発、販売する数理証明ツール

SAL:SRI International が開発している、検証ツール、証明ツールなどを接続するフレームワーク。この場合、SALの下に3種類のモデル検証器がつながっている。

Communication of the ACM:VOL 53 No2等の内容を参照しとりまとめ

米国における形式手法の適用事例:Rockwell Collins(2/2)

- Rockwell Collins社おける形式手法の事例(続き)
  - 事例:ADGS-2100 Adaptive Display and Guidance System Windows manager(WM)

    ヘッドダウンおよびヘッドアップディスプレイの管理ソフトウェアであるWMをSimulinkにより、ブーリアンと列挙型により定義し、NuSMVのようなBDDベース(二分決定グラフ:Binary Decision Diagram)のモデル検査器での検証に最適化した。

    WMは5つの主コンポーネントからなり、それぞれを独立に分析することができる。5つのコンポーネントは合計、16,117個のSimulinkのプリミティブブロックからなり、4,295個のSimulinkのサブシステムのインスタンスにグループ化された。

    5つのコンポーネントの可達空間(Reachable State space)は、$9.8X10^{9}$から$1.5X10^{37}$ステートであった。最終的に563のプロパティが開発、検証された結果、98の不具合がWMモデルの初期版で修正された。

  - 事例:CerTA FCS Phase 1

    米国空軍研究所の委託による先進航空重要システムの認定技術の研究プログラムにおいて、モデル検証とテストの研究が行われた。Lockheed Martin により開発された無人航空機の運用航空プログラム(OFP)にRockwell Collinsのツールを適用した。OFPは、適応性航空制御システムで、フライトの状況に応じて飛行のふるまいを変更制御する。

    開発の早い段階における検証により、テストでは発見することが難しい、断続的にほぼ同時に起こる、あるいは複合的な組合せの不具合を発見することができた。テスト段階で不具合を発見することにより、コスト面での効果があった。

  - 事例:CerTA FCS Phase 2

    モデル検証が数値的に非常に大規模なモデルに適応できるかを実証した。OFPの「Effector Blender Logic」のプロパティが、モデル検証器とトランスレータフレームワークで検証された。

    浮動小数点処理と大規模なステート領域のためNuSMVのようなBDDベースのモデル検査では検査が難しいため、Prover Technology のSMTソルバーを利用した。しかしながら、浮動小数点処理の扱いが難しく、検出の精度に課題があった。

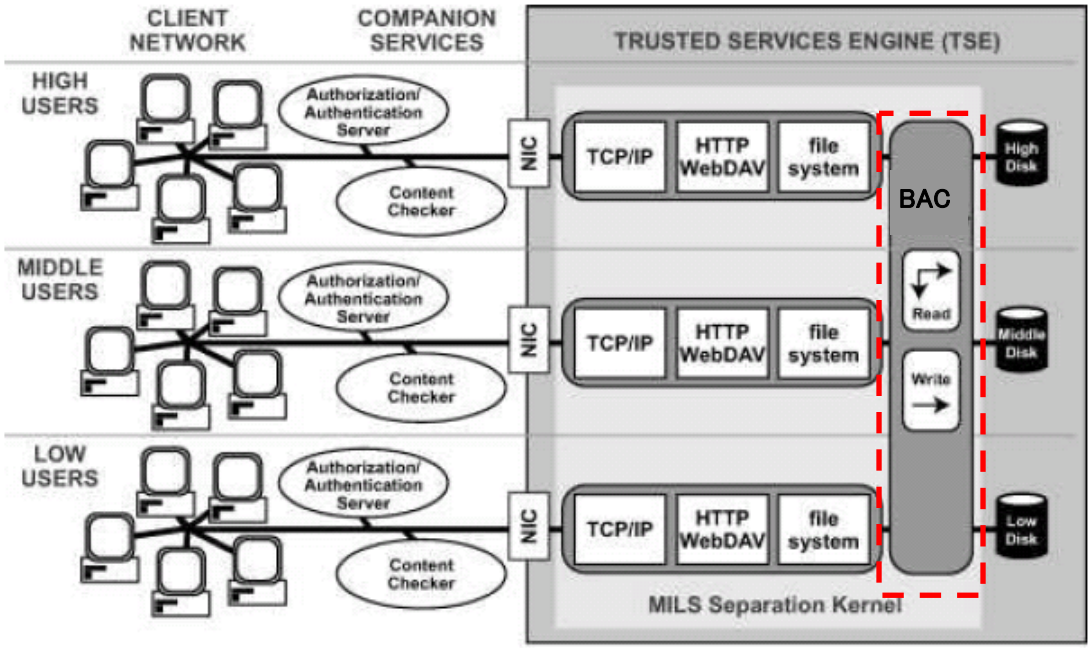
Communication of the ACM:VOL 53 No2等の内容を参照しとりまとめ

米国における形式手法の適用事例:Galois(1/2)

- Galois社による形式手法の事例
  - 事例:Trusted Services Engine(TSE)

ネットワークソフトウェアアプライアンスであり、複数のセキュリティレベルでのファイルの共有を可能にする。高いセキュリティレベルの権限のあるユーザは、統合したビューでのアクセスを可能とし、低いセキュリティレベルの権限のユーザのファイルのアクセスを可能とする。TSEは、EAL6の認証基準に基づき設計された。

TSEは、宇宙・海洋作戦システム・コマンド(Space and Naval Warfare Systems Command; SPAWAR)とNSAによりファンドされた、米国政府のオフザシェルフソフトウェア開発プロジェクトで、EAL6のセキュリティ基準に基づき開発された。TSEのブロックアクセスコントローラの開発に形式手法が適用された。



TSEのBAC(ブロックアクセスコントローラ)(注1)は、780行のCコードで記述されており、下記の手順で検証された。

1. コードの形式モデルの開発
2. コードに対応するモデルの検証
3. ポリシの形式モデルの開発
4. コードがポリシを実装していることを確認するためにモデルベースのテストを採用した。
5. コードがポリシを実装していることを形式検証した。

米国における形式手法の適用事例：Galois(2/2)

形式検証では、TSEセキュリティポリシマップが、直接モデル化され、そのモデルが実装された。この際に Isabelle Higher Order Logic（HOL）定理証明(注2)が利用された。

モデルをマップし、実装するために、「code-to-spec review」レビューチームが、最低2名で、HOLコードとC実装のラインごとの検査を実施した。

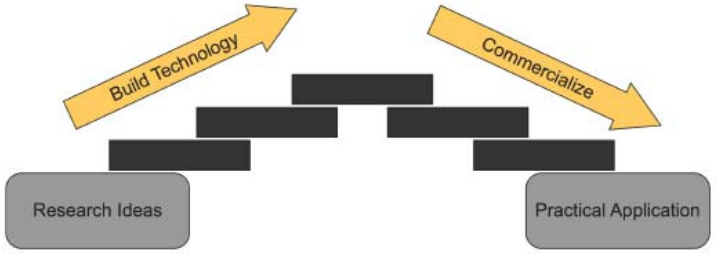
モデルベースのテストの取組みでは、 QuickCheckツール(注3)を利用した。セキュリティポリシの形式記述に基づき、QuickCheckはテストケースを生成し、実装がポリシに違反していないことを確認した。

注1)BAC(ブロックアクセスコントローラ)：高信頼性のクロスドメインのソフトウェアコンポーネントとして、複数のセキュリティレベルのアクセスを調整し、高いセキュリティ権限のアクセスには、読み／書きのアクセスを許諾し、低いセキュリティ権限のアクセスには、読込みだけを許可する

注2)Isabelle Higher Order Logic(HOL)：高階述語論理(High Order Logic)による定理証明

注3)QuickCheck：QuviQ社(スウェーデン)が開発、販売するテストツール

- Galiois社について
  - 設立：1999年
  - 従業員数：約40名
  - 所在地：Portland Oregon(米)
  - 大学の機能プログラミングと形式手法の研究のスピンアウト企業として、高信頼性ソフトウェア技術関連の研究開発の受託、製品開発、技術移転を行う。
  - 主な顧客は、DoD



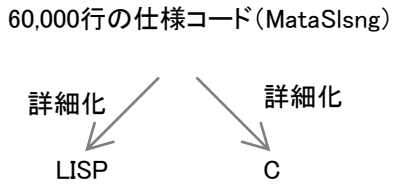
米国における形式手法の適用事例:Kestrel Technology

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

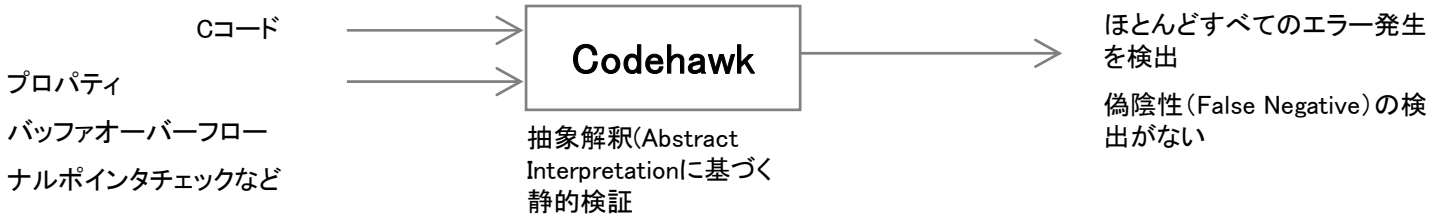
- Kestrel Technology社による形式手法の事例
  - 事例:Javaスマートカード

    米国政府の委託を受け、ATMEL Javaスマートカードの開発を行っている。スマートカードの仕様は、MetaSlang言語で表現され、LISPとCに変換される。EAL7の基準に沿って開発されている。開発は Kestrel Technology社の開発者3名(PhD)により、2005年から行われており、現在も継続中。証明ツールにはIsabelleが利用されている。

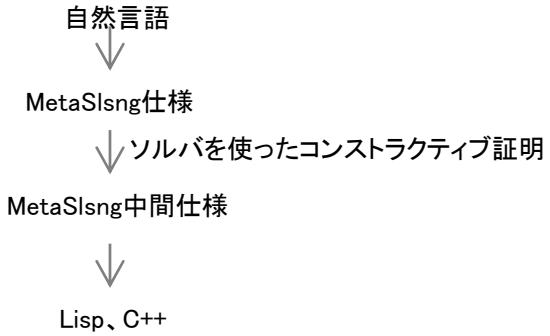
    MetaSlsmgは Kestrel Instituteにより開発されている形式手法表記である。



- Kestrel Technologyについて
  - 米国のカリフォルニア州サンフランシスコ郊外にある Palo Alto にある従業員6名程度の研究開発企業。Kestrel Instituteから分離独立して、2000年に設立された。従業員の多くはスタンフォード大学など形式手法の研究が進んでいる大学出身のPhDであり、主に、米国国防総省のソフトウェア開発を受注している。Kestrel TechnologyではCコードの静的検証ツールであるCodeHawkを開発している。



- Kestrel Institute について
  - Kestrel Institute は、Kestrel Technologyと同じ建物内にある非営利の開発組織であり、1981年のCordell Green氏により設立された。10名程度の研究者がいる。形式手法ツールである Specware を開発している。



米国における形式手法の適用事例：その他

SEC
Software Engineering
for Mo・No・Zu・Ku・Ri

- その他の動向
  - AADL(アーキテクチャ記述言語)

    SAE(米国自動車技術協会;Society of Automotive Engineers)が推進するアーキテクチャ記述言語であるAADLをサポートするオープンソースツールであるOSATEにより、モデルベース設計や形式手法のツールをプラグインすることができる
  - マサチューセッツ工科大学(MIT)

    MITのコンピュータサイエンス学部、Daniel Jackson 教授の下、B と Z の合金という意味で作られた新形式言語、Alloyと呼ばれる形式手法表記を開発がされている。英国のグランドチャレンジプログラムの Verified Softwareに参加し、既存のオープンソースプログラムのAlloyによる再検証を実施している。AT&T研究所によるAlloyを利用したネットワークプロトコルの研究などの研究に利用されている。Rockwell Collins では、AADLでアーキテクチャを定義し、Simulink でモデル化したシステムをAlloyを使い形式検証することで、良い結果を得た例があるという。Alloyのコミュニティに実事例についての問合せをしているが、現在までに有効な回答は得られていない。
  - ニューヨーク大学

    コンピュータサイエンス学部のClark Barrett教授の下、Cソースコードをより表現力の高い言語に変換し、SMTソルバで解析するツール、Cascadeを開発している。
  - NEC北米研究所

    LSIの設計検証のモデル検査エンジンをソースコードの検証に適用する研究を行っている。その成果として、形式手法によるC言語検証ツール「VAREL」(NEC社内で利用されており、外販はされていない)がある。
  - マイクロソフト

    C#の拡張言語であるSpec#の開発がされている。Spec# Static Program VerifierがSpec#プログラムの検証を行う。
  - ソフトウェア検証ツール、テストツール

    ソフトウェアの検証ツールベンダがCコードの検証ツールの市場を形成している。Coverity(San Francisco, CA)、Klocwork(Burlington, MA)、Parasoft(Monrovia,CA) 、Bsquare(Bellevue, WA)等
  - ハードウェア検証ツール

    ハードウェアの形式手法検証ツールがEDAベンダにより提供されている。Synopsys(Formality)、Cadence(Incisive Formal Verifier)、Mentor(0-In Formal Verification)やJasper、Calypto、 Real Intent、Atrentなどのスタートアップ企業

欧州と米国における形式手法へのアプローチの相違

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- 欧州では、形式手法による表記とその検証、証明を行う取組みが多い
- 米国ではモデルベースの設計と検証、ソースコードの検証を形式手法(論理証明)で行う取組みが多い。またHWの形式手法検証(ベリフィケーション)ツールを提供するツールベンダの実績がある

設計レベル		記述例		ツール					
				設計		検証、証明等		変換	
SW	HW	SW	HW	SW	HW	SW	HW	SW	HW
アーキテクチャ設計		AADL、SysML		モデリングツール（OSATE等）		シミュレータ（ADeS等）		アーキテクチャからSWコード生成	
システム設計		Simulink等		システム設計ツール（Simulink等）		シミュレータ(Simulink等)検証（Simulink Design Verifier等）		システムモデルからSWコード生成	システムモデルからHDL生成
		B、Scade、Z、VDM、CSP、SPARK等		統合形式手法ツール（AtelierB、SCADE、Spark等）		モデル検証ツール、論理証明ツール、証明責務ツール		形式仕様からSWコード生成	
ESL(Electronic System Level)設計		C、C++、Ada、Java、アセンブラ等	SystemC、ツール独自HDL	ESLツール（VaST、コウェア等）		ESLツール（VaST、コウェア等）			モデルからHDL生成
プロトタイプ実装		C、C++、Ada、Java、アセンブラ等	（プロトタイプ）	プロトタイプ設計ツール（dSPACE等）		静的、動的チェック、証明ツール	ハードウェアシミュレータ		
SW実装	HW実装	C、C++、Ada、Java等	VHDL、VerilogHDL（RTL、ゲート）	SW設計ツール	回路設計ツール	静的、動的チェック、証明ツール	静的、動的チェック、HW検証ツール、シミュレータ	ソースコードのコンパイル	
コンパイル	論理合成	ランタイム	回路	コンパイラ	論理合成	ICE	ロジックアナライザ		HDLから回路を論理合成

米国的アプローチ　欧州的アプローチ

欧州調査と米国調査の知見

SEC
Software Engineering
for Mo·No·Zu·Ku·Ri

- 欧州調査で得られた知見
  - 2000年以降、BやSCADE等のツールベンダ出現とともに、実応用事例が急激に増えている(欧州の研究開発プロジェクトが形式手法の実応用化の基盤)
  - 形式手法は航空宇宙、鉄道等大規模システムへの適用が多い
  - 最近ではスマートカード、RTOS、フラッシュメモリドライバ等小規模システムへの適用が増えてきている
  - 実応用への適用ではBとSCADEの比率が高い(Bは鉄道システムへの実応用、SCADEは原子力、航空機への実応用で成功を収めた)
  - 自動コード生成機能を持ったツールが多く利用されている(B、SCADE等)
  - 自動コード生成を利用しない場合は、形式手法の適用により全体工数は増加傾向にあるが、自動コード生成を利用した場合は増加傾向は見られない(全体工数が減少したとの報告もある)
  - 自動コード生成を利用した場合は上流工程(仕様作成、詳細化等)の工数比率が高くなっている
  - 仕様作成、詳細化には、ドメインの知識に加え表記の記述、読解のスキルが必要になるが、コード化(実装)、テスト、妥当性検証では、読解ができれば十分
  - 基礎的な学力と社内の研修システムがあれば、形式手法の未経験者であってもプロジェクトに参画できている
  - SaaS方式でツールを提供するツールベンダが出現している(医療機器、半導体製造装置などへの適用事例、サブシステムサプライヤでの利用事例がある)
  - 国際標準で形式手法利用を推奨する例が増えている
- 米国調査で得られた知見
  - 欧州では、形式手法表記(Z、B、CSP等)により仕様を記述し、検証を繰返しをしながら、実装コードまで詳細化する方法が多く、米国では、Simulinkなどのモデルベース設計ツールで仕様を入力し、その検証過程で形式手法による検証、証明をしていく方法が多い
  - 実装コード(CやC++)の静的チェックをするツールベンダ(Coverity等)が多くあり、SMTソルバによりコードを検証する研究も多い。
  - マイクロソフトは、C#の拡張言語であるSpec#を開発、プログラム検証ツールとして、Spec# Static Program Verifierがある。
  - NEC北米研究所やニューヨーク大学など、ソースコードをモデル化し、検証(SMTソルバ)していく研究がされている。
  - 欧州では、EUプロジェクトやオープンソースとしての形式手法ツールや表記の開発が進んでいるが、米国では、企業内のプロプライエタリツールが開発される場合がある。
  - EDAベンダがハードウェア検証ツールを開発し、実用化している。
  - 米国では、セキュリティ標準(EAL)に準拠するために形式手法を利用する例がある。

IPA Software Engineering Center
Information-technology Promotion Agency, Japan

2010年7月29日

独立行政法人　情報処理推進機構
ソフトウェア・エンジニアリング・センター