

# QN-GENの拡張機能開発における 形式手法の適用事例

○三善 浩司<sup>\*1</sup>, 中城 亮祐<sup>\*1</sup>, 三輪 直樹<sup>\*1</sup>, 平田 将乙<sup>\*1</sup>,  
内藤 充昭<sup>\*2</sup>, 岩崎 孝司<sup>\*2</sup>, 日下部 雄三<sup>\*2</sup>, 井上 康生<sup>\*2</sup>,  
荒木 啓二郎<sup>\*3</sup>, 日下部 茂<sup>\*3</sup>, 大森 洋一<sup>\*3</sup>

\*1 九州大学大学院 システム情報科学府

\*2 富士通九州ネットワークテクノロジーズ株式会社

\*3 九州大学大学院 システム情報科学研究院

# 発表の流れ

---

- 導入
- プロジェクト実施内容
- まとめ

# 導入

---

- 背景
- プロジェクト概要
- プロジェクトの背景
- QN-GENとは?
- ドメイン特化型開発
- 形式手法


# 背景：教育面

---

- 産学官連携によるICT人材の教育
  - 平成18年5月日本経団連「高度情報通信人材育成プロジェクト」の推進拠点に選定
  - 平成18年9月文部科学省「先導的ITスペシャリスト人材育成推進プログラム」採択（平成21年度まで）
  
- 先進的・多角的なICT実践力を重視したカリキュラム
  - オムニバス形式の講義
  - 長期インターンシップ
  - **PBL(Project Based Learning)による実践的教育**

# 背景：技術面

---

- SDN (Software-Defined Networking)
    - ネットワークの構成や機能の設定をソフトウェアによってプログラマブルに行える
    - ソフトウェア開発の問題
- 
- ソフトウェア工学的なアプローチが必要になる...?
    - 形式手法
    - ドメイン特化型開発
    - モデル駆動開発
    - ...

# プロジェクト概要

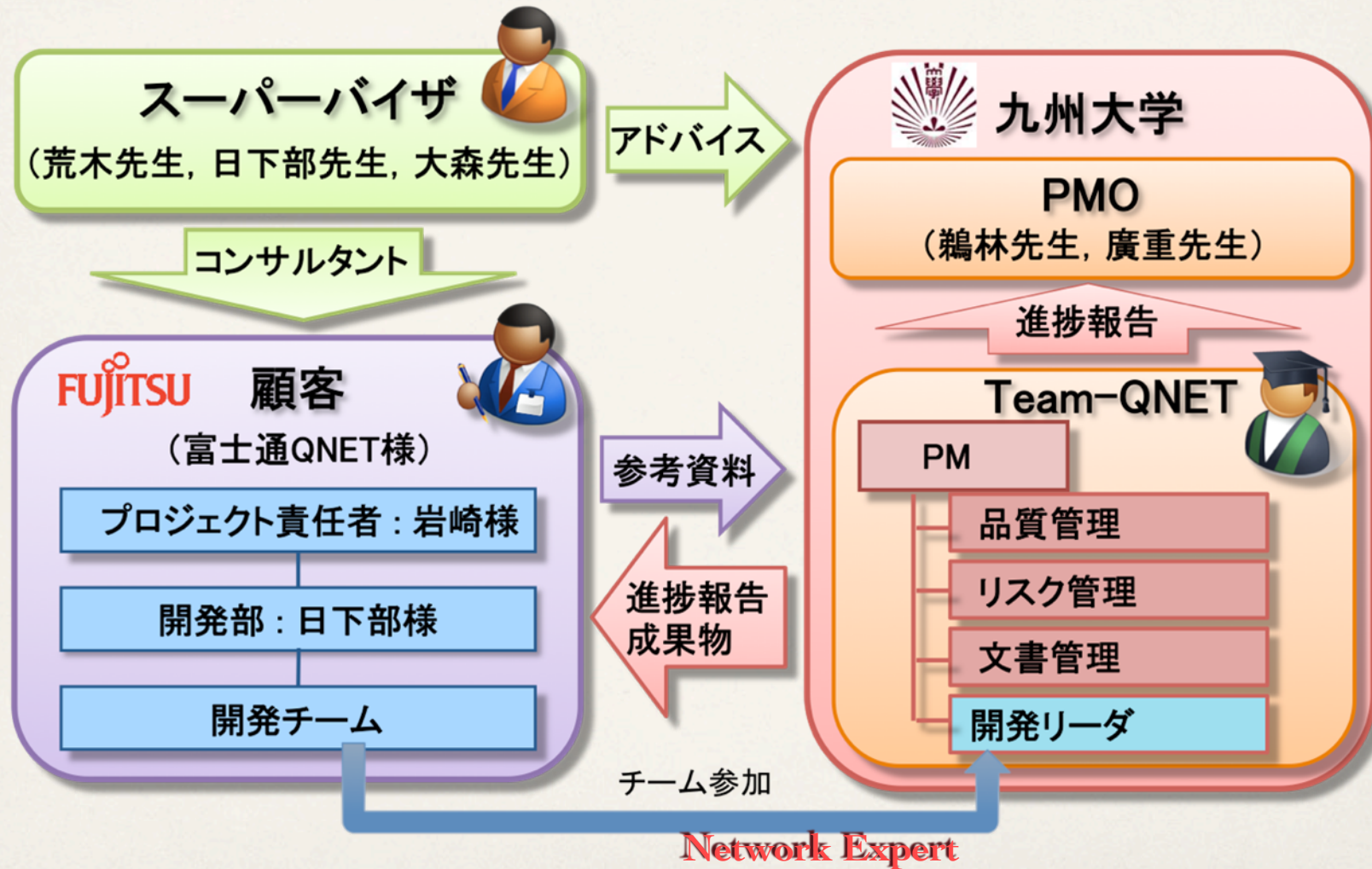
---

---

## -形式手法を用いたOpenFlow向DSL開発プロジェクト-

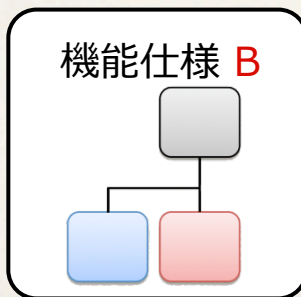
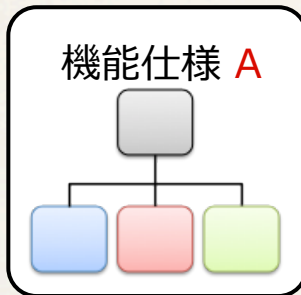
- 顧客
  - 富士通九州ネットワークテクノロジーズ株式会社
- 実施期間
  - 2011年10月17日～2012年2月21日(PBL2),
  - 2012年 4月17日～2012年7月20日(PBL3)
- 体制...
- 実施内容
  - QN-GENシステムの拡張開発ならびに品質確保
  - 形式手法を導入したDSL開発プロセスの知見収集

# ステークホルダ



# QN-GENとは？

- QN-GEN : QNET Network control system generation platform
  - OpenFlowを用いてネットワーク制御を行うソースコードを自動生成するプラットフォーム
  - 独自に定義したモデリング言語によって記述されたモデルを入力にあてる



入力：モデル図  
出力：ソースコード



# プロジェクトの背景

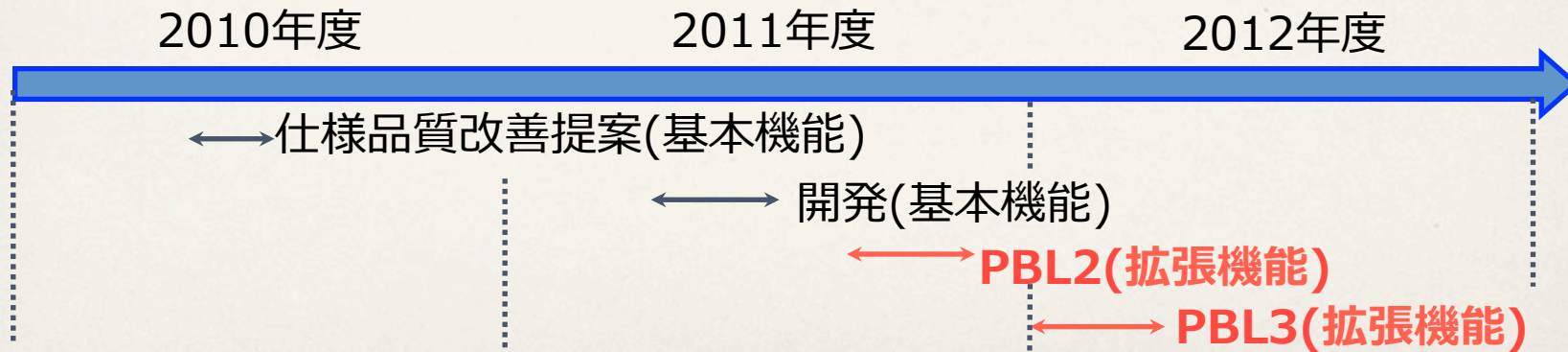
- QN-GENの開発は段階的に実施される
  - 基本機能, **拡張機能**, 将来機能  
(経路設定, **統計情報収集**, 構成管理など)
- PBLのテーマとしては2年目
  - 2010年度のPBLで基本機能に関して仕様品質改善提案が行われ, 顧客側にて開発も完了
  - 2011年度は**拡張機能**を対象に, 仕様・設計品質改善と開発に取り組む

## QN-GEN

基本機能(経路設定)

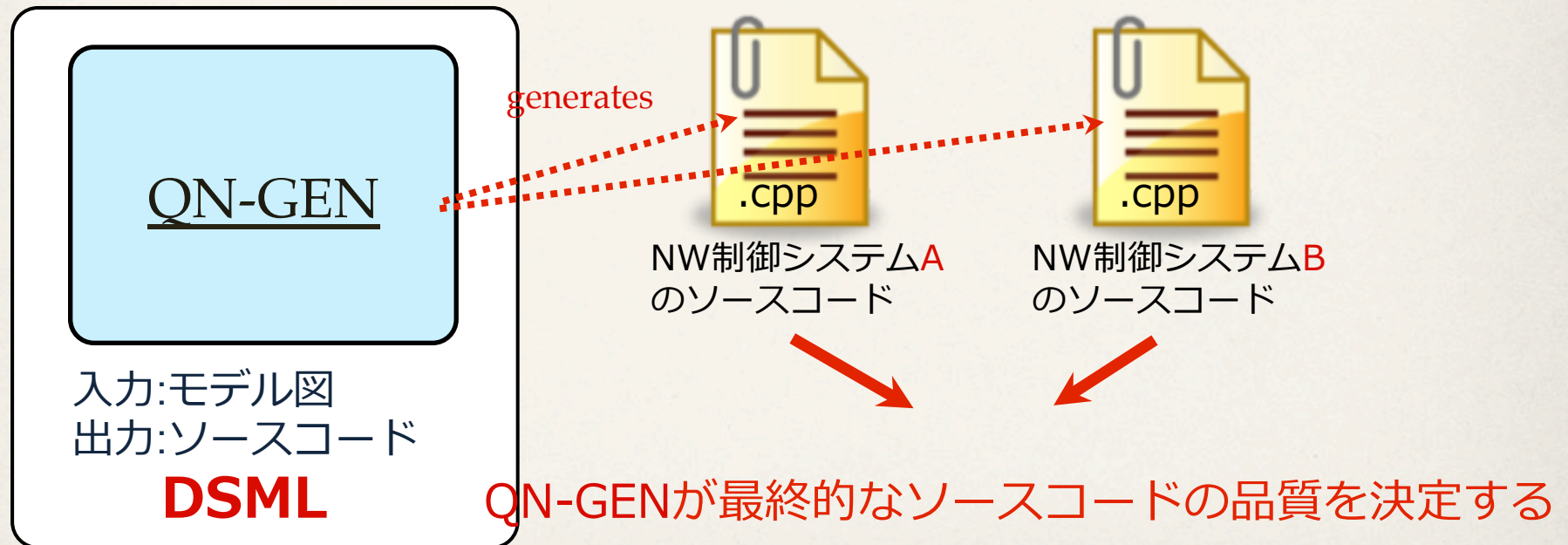
**拡張機能(統計情報収集)**

将来機能(構成管理等)



# QN-GEN as DSML

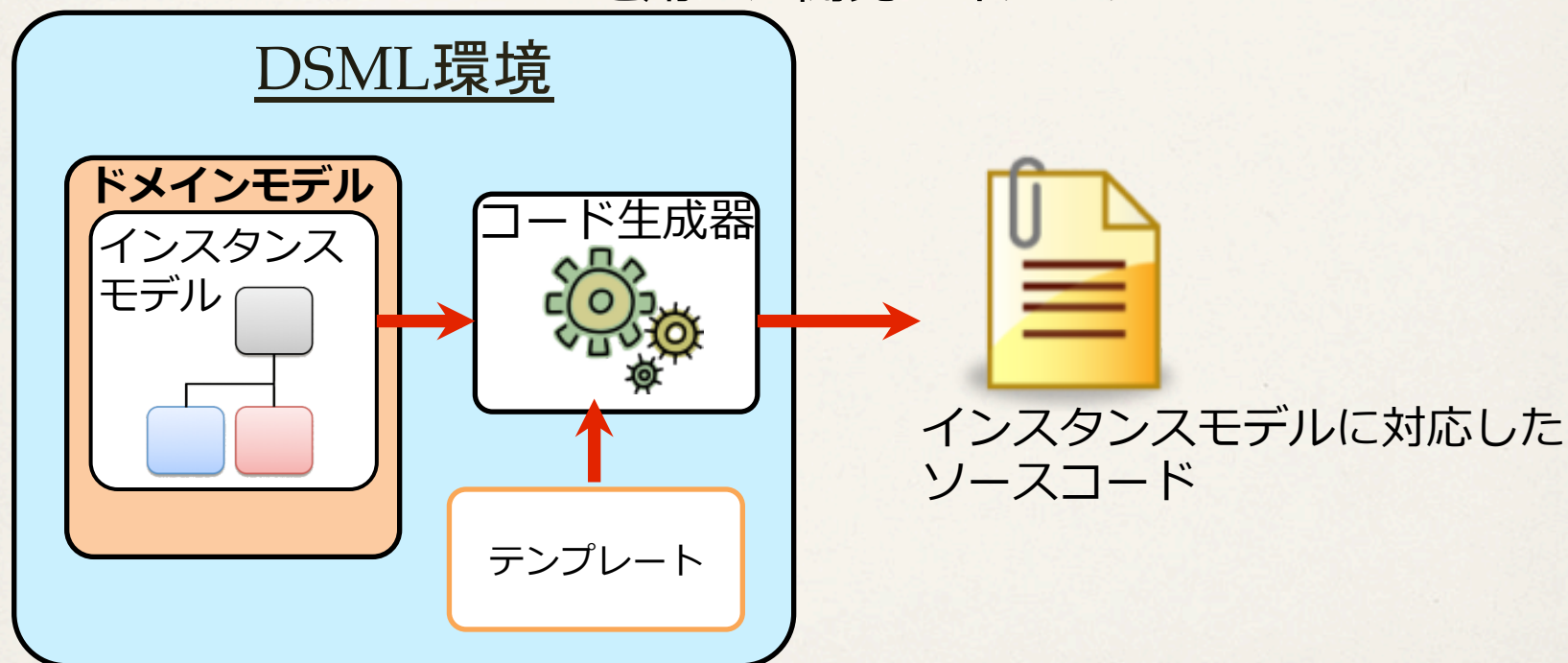
- DSL : Domain-Specific Language
  - 特定の問題領域に焦点を当てることで、汎用型言語よりも生産性,  
信頼性, 再利用性などの向上を目的とする言語
- DSML - グラフィカル形式に記述するDSL



# ドメイン特化型開発

- 特定目的向けの言語を設計することによってソフトウェア開発の生産性や品質を向上させるソフトウェア開発方法論

DSMLを用いた開発のイメージ



# 形式手法

---

- 数理論理学による裏付けをもった言語を用いるシステム開発手法
  - 厳密な仕様を記述したり, システムの注目する性質について検証を行うことができる
- 代表的な分類
  - **モデル規範型**: 内部の構造や, 状態の変化をモデル化する  
不変条件, 事前条件, 事後条件を記述
  - 性質規範型: システムを外部から見て, その性質を公理や抽象データ型を用いて表現する
  - **モデル検査**: 状態遷移モデルとモデルが満たす条件を, 振る舞い仕様として記述して, 全状態空間を生成し, 自動検査する

# プロジェクト実施内容

---

- プロジェクト全体像
- 自動生成環境(QN-GEN)の拡張開発
  - 形式手法の適用
    - VDM, Alloy, SPIN
- 自動生成環境(QN-GEN)を利用した開発

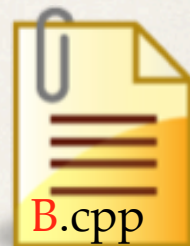
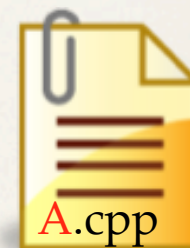
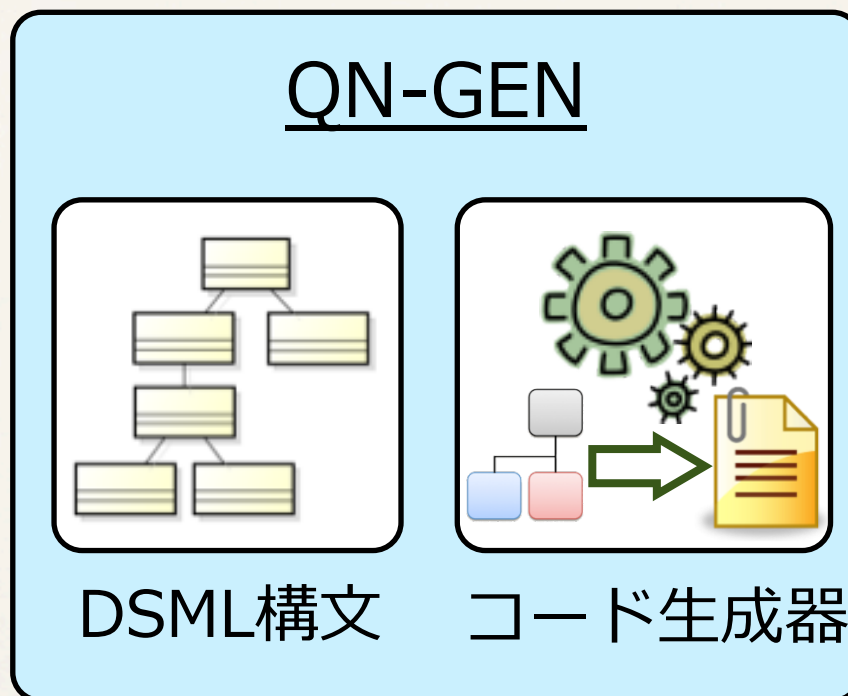
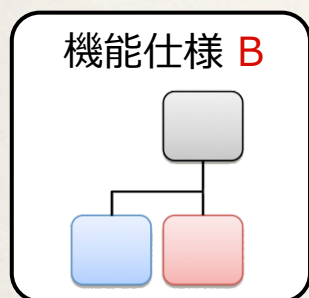
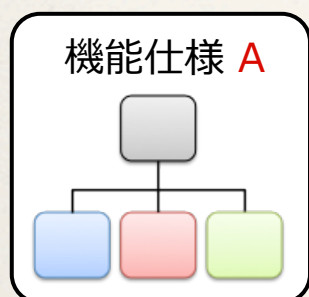
# DSML開発について



- 対象領域の特性、機能を分析する
- DSMLの構文を定義する
- 自動生成の枠組みを作成する
- 生成対象をモデル化する

# DSML環境(QN-GEN)の構築

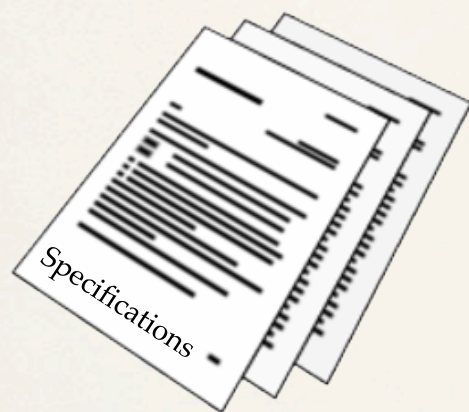
高い確信度で構築するには？



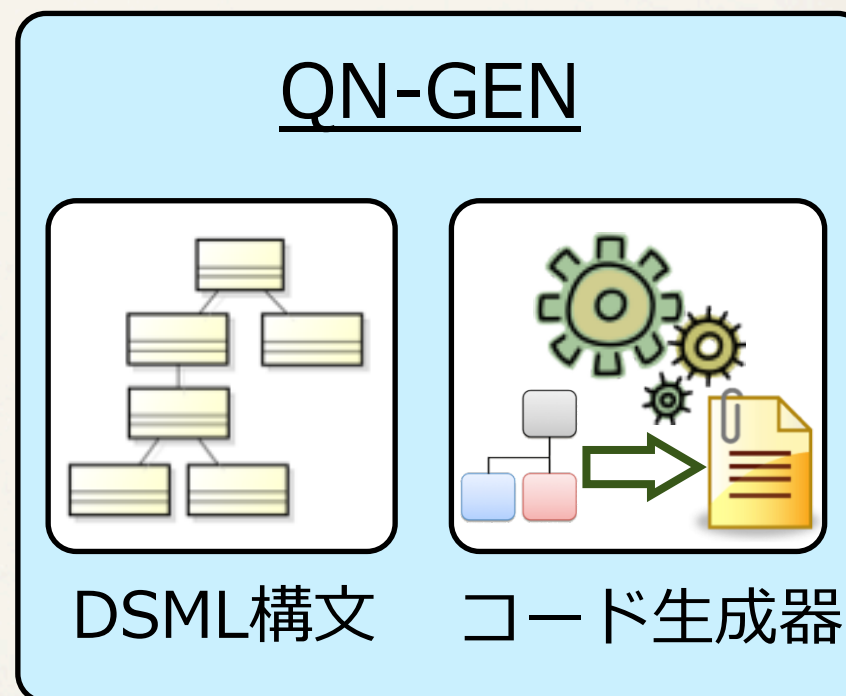
入力：モデル図  
出力：ソースコード

# DSML環境の構築

高い確信度で構築するには？



外部仕様書



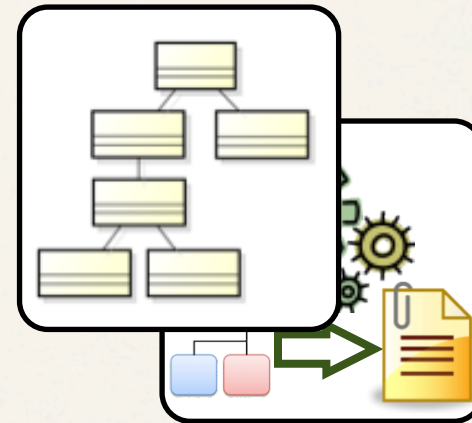


# 品質管理

- ソースコードだけでなく、システム仕様・設計成果物の品質を問う



システム仕様



設計成果物

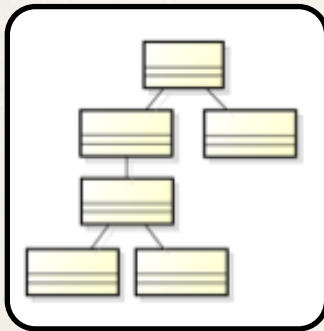
どんな品質特性が重要なのか？  
どのように品質を確保するのか？

# 品質管理



## システム仕様

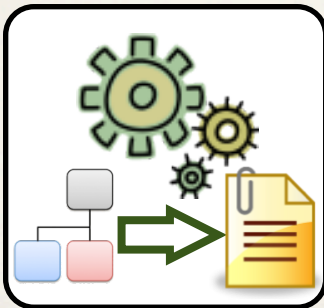
- ・ 厳密性 : 形式仕様記述の曖昧さ・抜け漏れがない.
- ・ 一貫性 : 仕様書と形式仕様記述が対応している.



## DSML構文

### 形式手法

機能を実現できる.

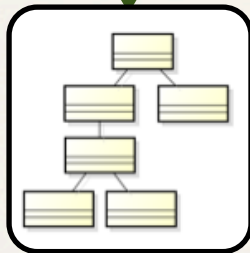
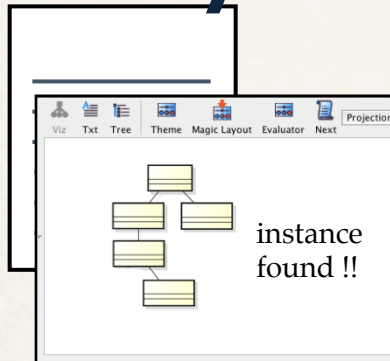


## コード生成器

- ・ 妥当性 : 生成すべきコードを誤り無く生成できる.
- ・ 一貫性 : DSML構文と対応したコード生成ができる.

# 適用する形式手法

## Alloy



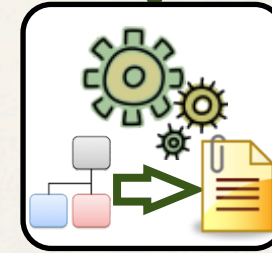
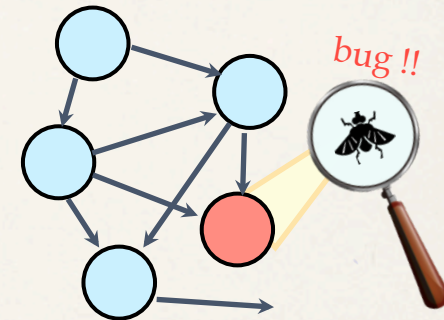
DSML構文

## VDM



システム仕様

## SPIN

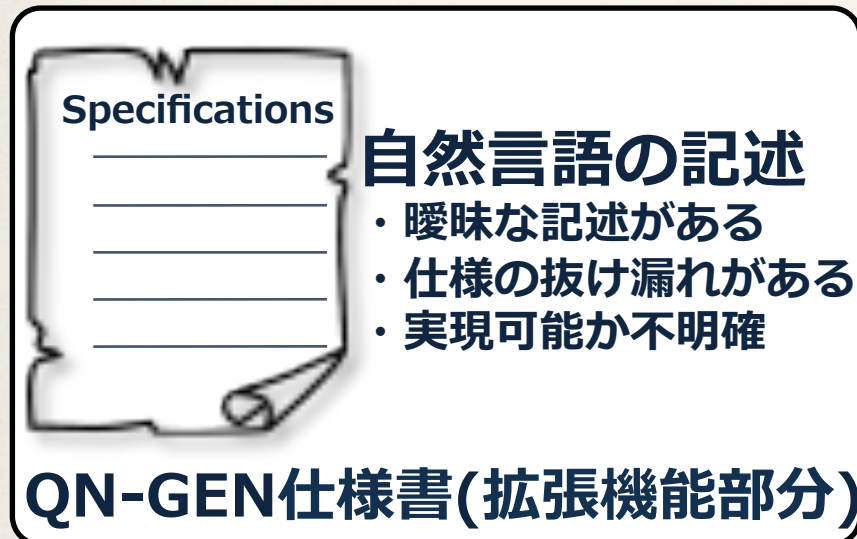


コード生成器

# VDM

## ~ Specification's Validation ~

- 仕様を厳密に記述していく過程で、仕様の曖昧さや抜け漏れを排除
- テストを通じて仕様の妥当性を確認



モデル化

```
class Controller
types
public Network = ...
...
operations
public Controller : ...
...
end Controller

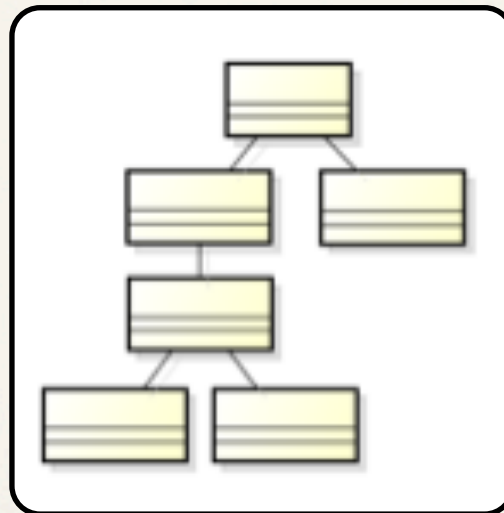
.vdmpp
```

**VDM 仕様記述**

# Alloy

## ~ Verification of Domain model ~

- 定義したDSML構文の表現力は正しいのか？
  - 文法の定義(要素間の関連や制約条件)から, その定義を満たすインスタンスを想定するのは容易ではない

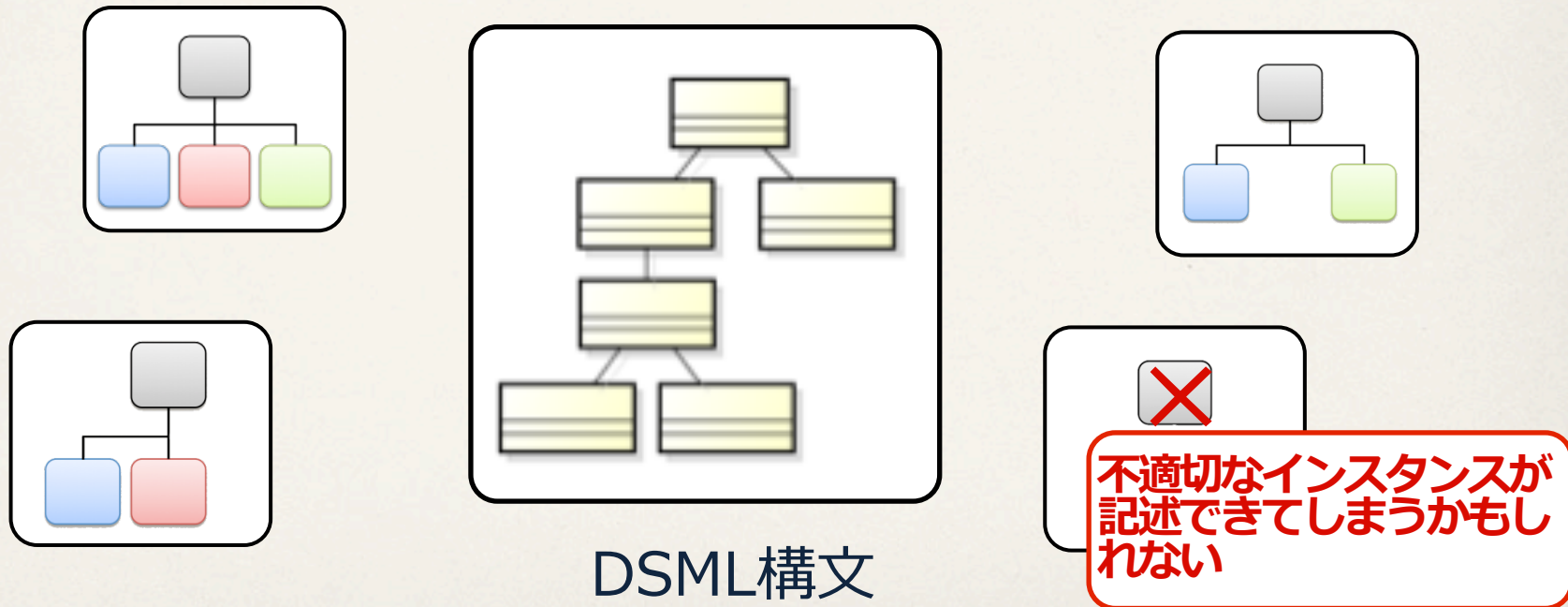


DSML構文

# Alloy

## ~ Verification of Domain model ~

- 定義したDSML構文の表現力は正しいのか？
  - 文法の定義(要素間の関連や制約条件)から, その定義を満たすインスタンスを想定するのは容易ではない



# Alloy

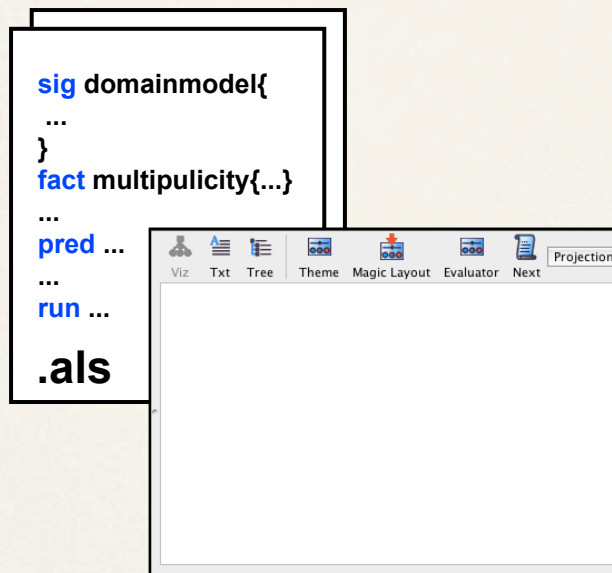
## ~ Verification of Domain model~

- DSML構文のモデル化および検証
  - 言語の文法そのものを Alloy で形式化し, 検証する

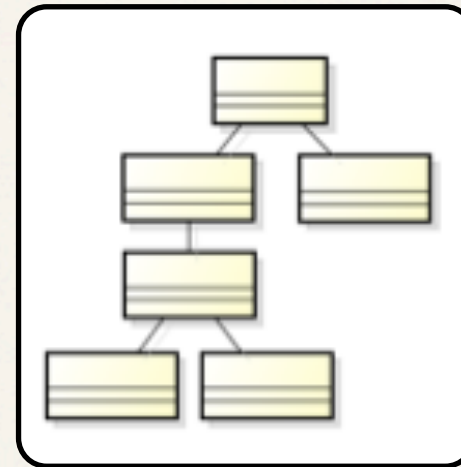


# よくわかる解説 ~ Alloy編 ~

## ~ Verification of Domain model ~



Alloy Analyzer



DSML構文



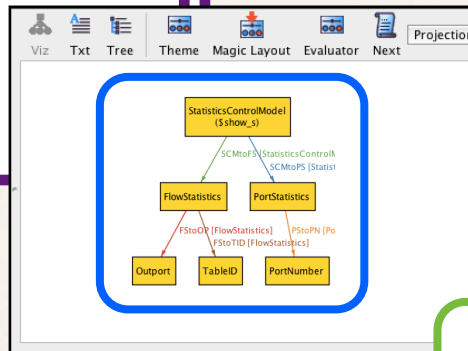
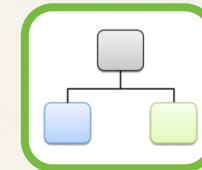
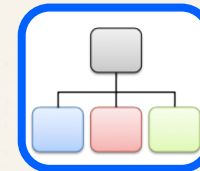
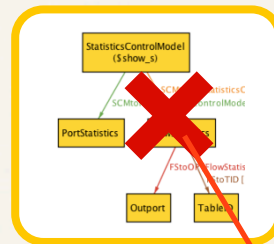
# よくわかる解説 ~ Alloy編 ~

## ~ Verification of Domain model ~

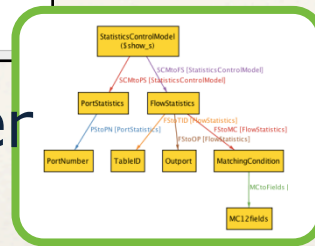
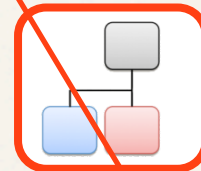
自動的かつ網羅的

人の手による確認

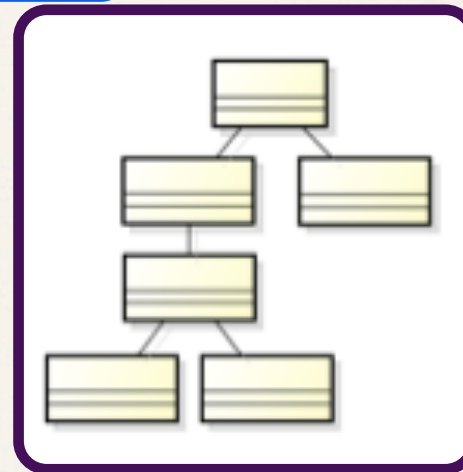
```
sig domainmodel{  
  ...  
}  
fact multiplicity{...}  
...  
pred ...  
...  
run ...  
.als
```



Alloy Analyzer



誤ったインスタンスの検出



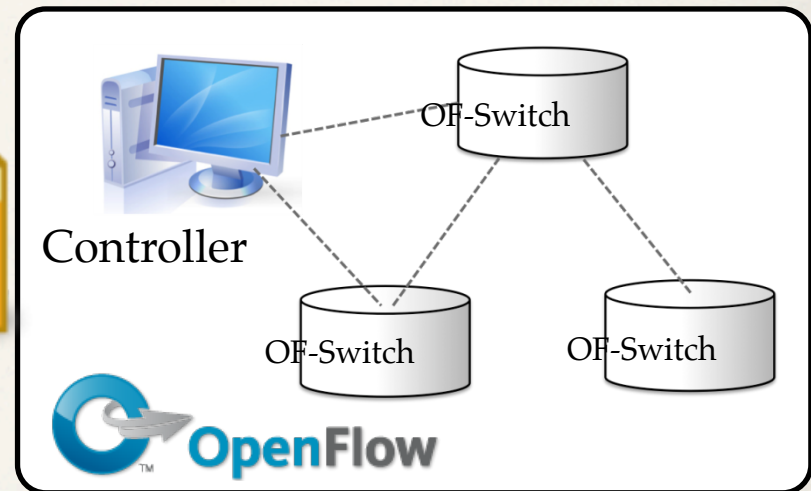
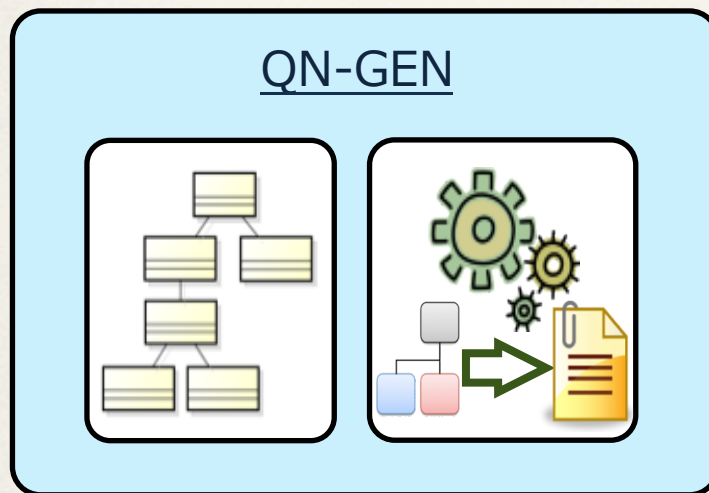
DSML構文



# SPIN

## ~ Verification of Network System Design ~

- 自動生成対象のシステムデザイン検証
  - 生成する実システムの設計検証を行う

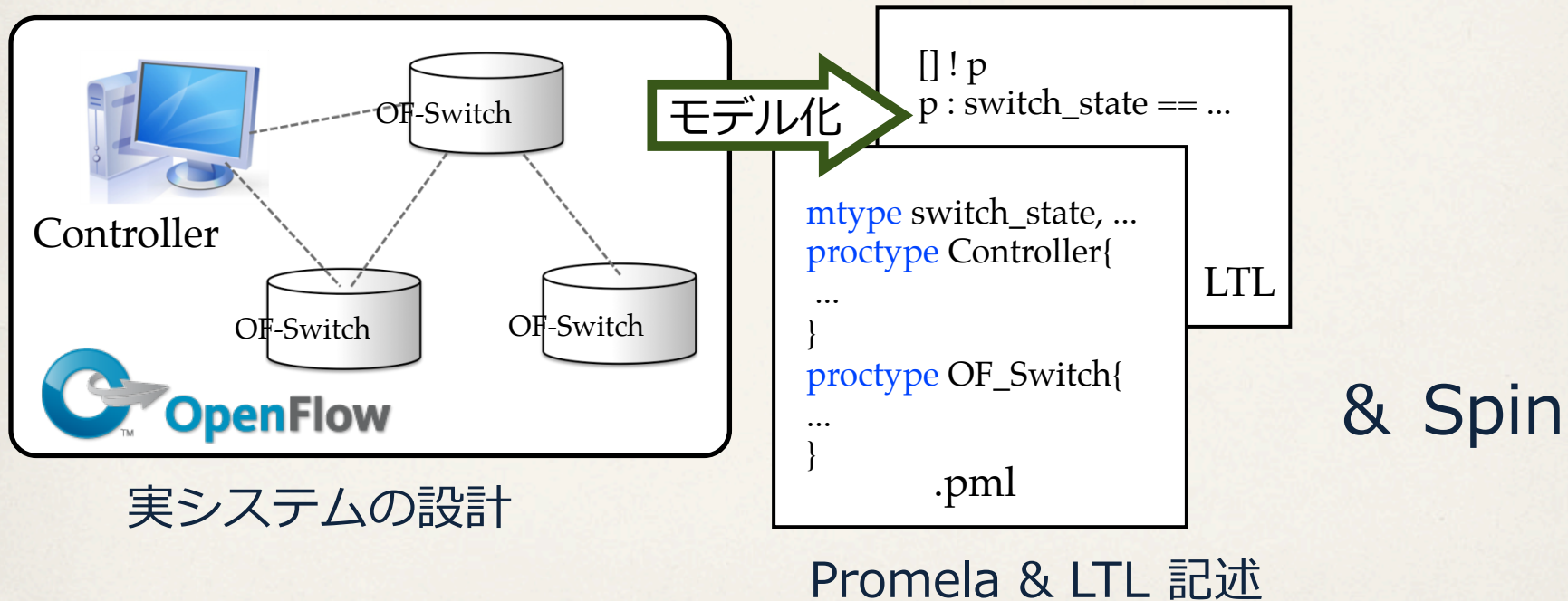


ネットワーク制御システム

# SPIN

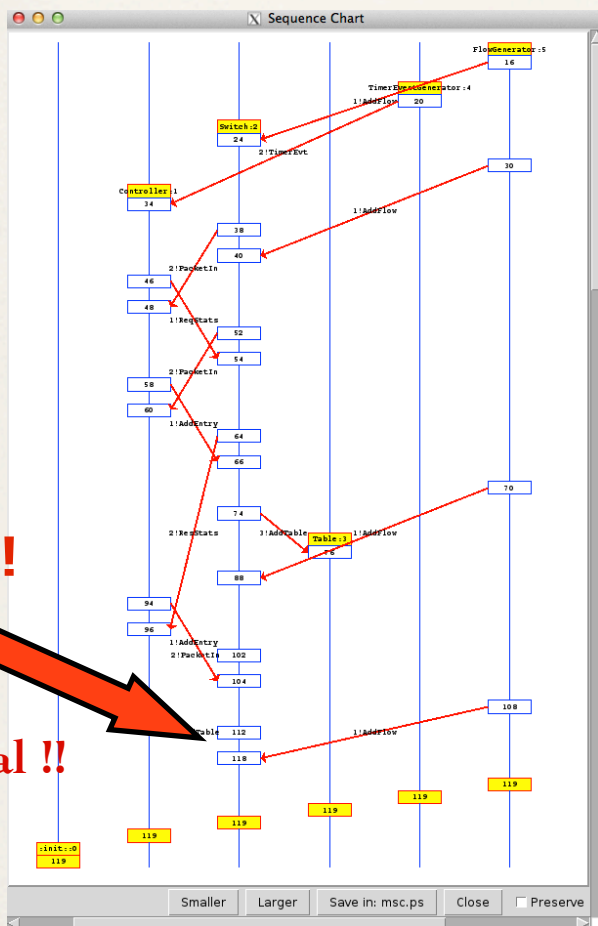
## ~ Verification of Network System Design ~

- 自動生成対象のシステムデザイン検証
  - 生成する実システムの設計検証を行う



# よくわかる解説 ~ SPIN編 ~

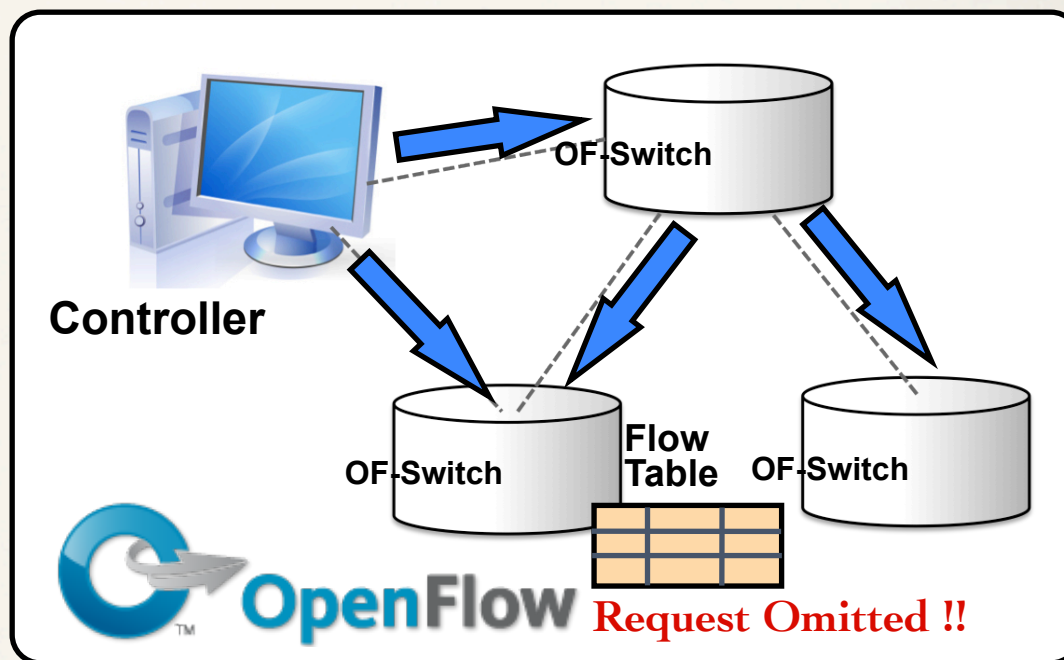
## ~ Verification of Network System Design ~



ココ!!

Critical!!

自動的かつ網羅的



あらゆるNW制御のパターンを網羅

# ここまでのまとめ

## 要求分析

要件定義

## DSML開発

フィーチャー分析

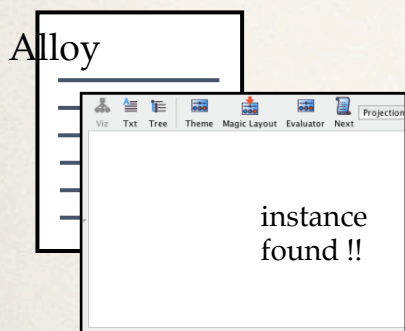
ドメインモデル設計

テンプレート実装

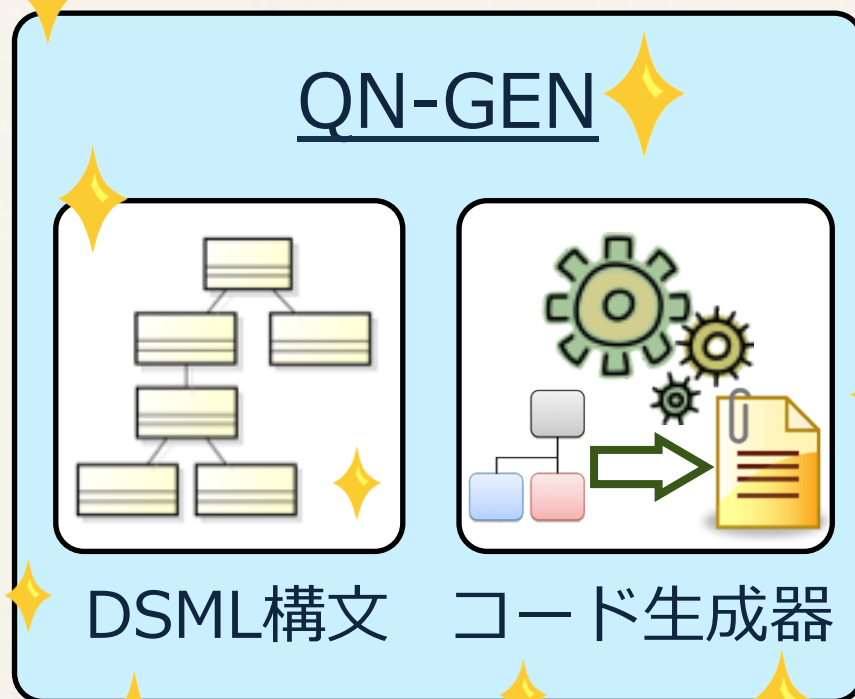
- VDM
  - 要件定義における仕様の曖昧性排除
  - 分析を通じて、対象の理解を深める
- Alloy
  - ドメインモデル（DSML構文）の妥当性を検証
- SPIN
  - 自動生成対象コードの設計の検証

# Formal QN-GEN

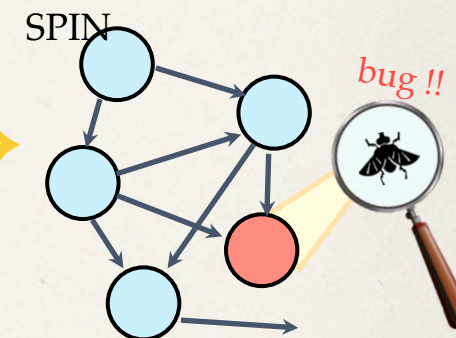
高い確信度！！



確信度の高いモデル！！



曖昧さを排除した  
厳密な仕様



自動生成後の  
システム設計検証

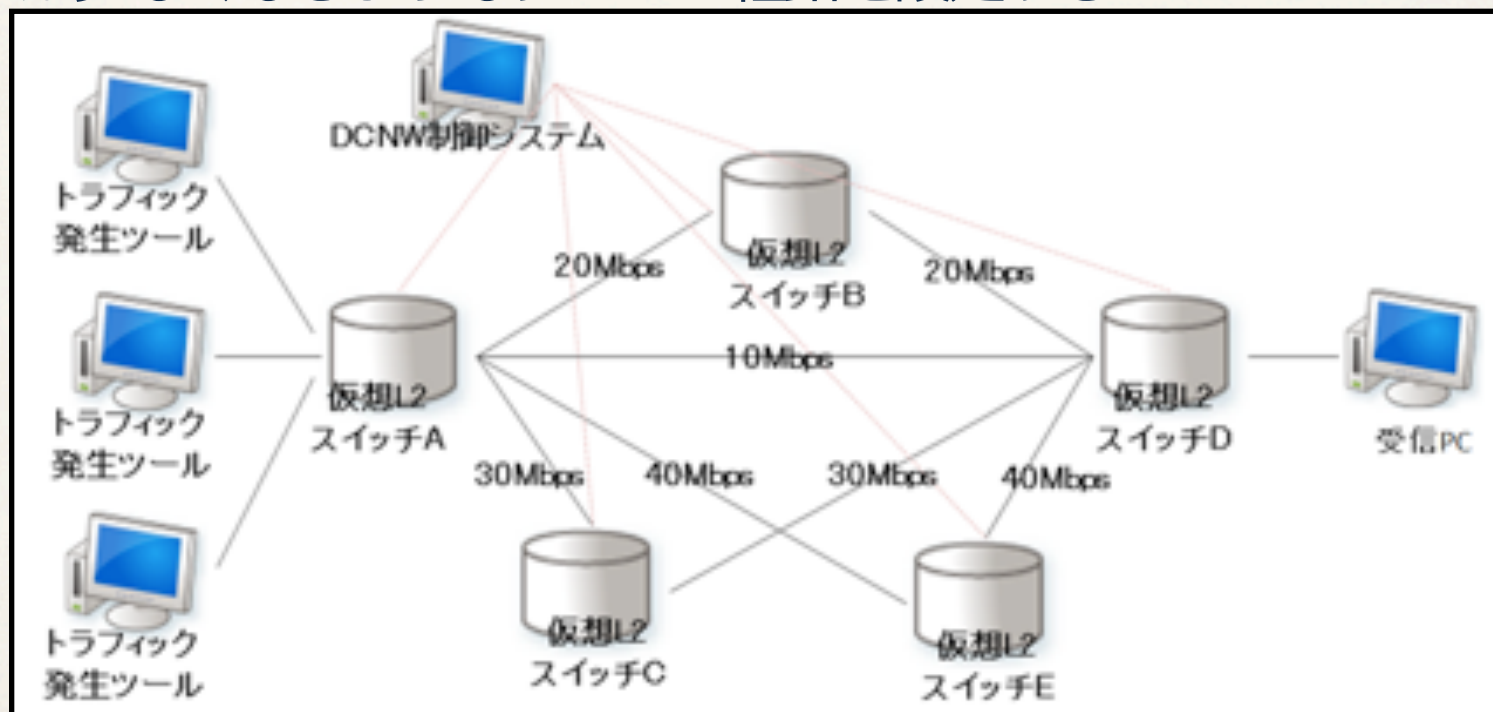
# DSML開発について(再掲)



- 対象領域の特性、機能を分析する
- DSMLの構文を定義する
- 自動生成の枠組み, 生成規則を作成する
- 生成対象をモデル化する

# 開発対象

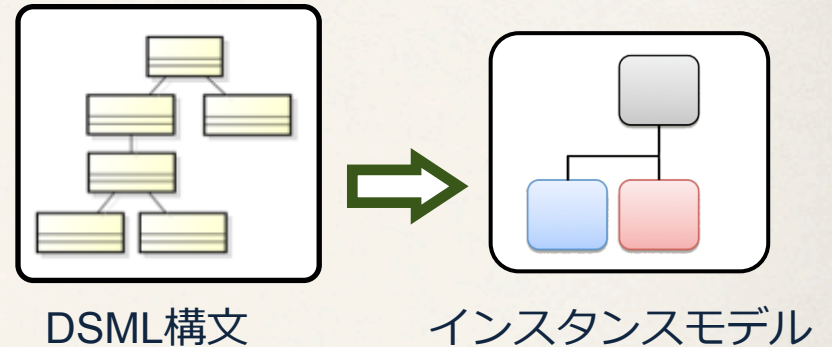
- スイッチ間にトラフィック量の上限が設定されている
- 一定周期で統計情報を収集する
- トラフィック量が上限を超えないように、かつ使用するスイッチが少なくなるようなフローの経路を設定する



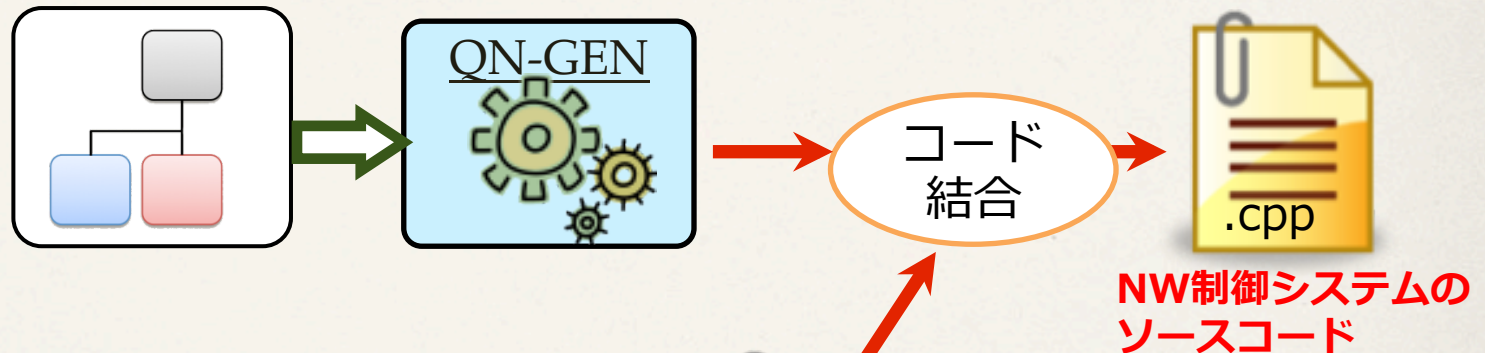


# QN-GENを利用した開発

- インスタンスモデル作成



- 自動生成



- 自動生成対象外コード作成

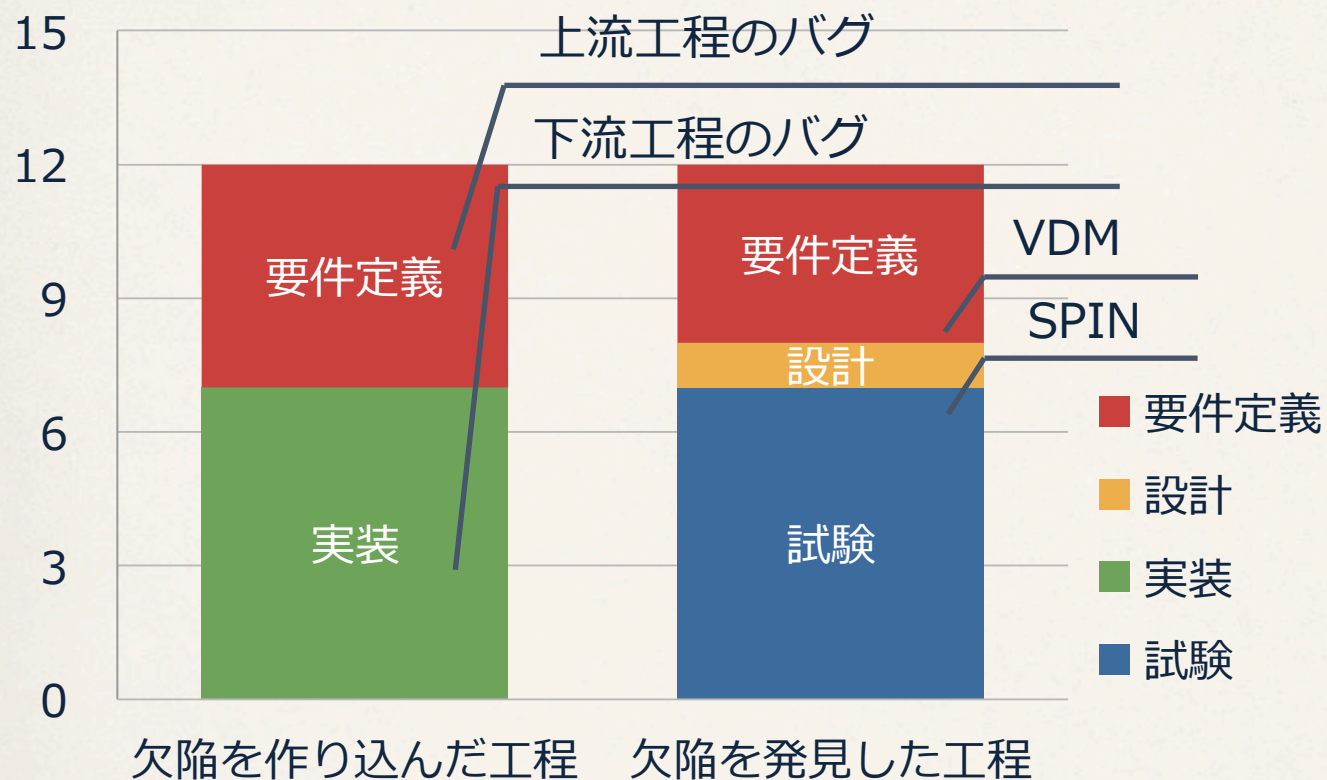


# プロジェクトに関するデータ

---

- 規模(ネットワーク制御システム)
  - 全体 : 2836 step
  - 自動生成部分 :  
1359 step(48%)
- 形式手法導入工数
  - VDM : 55h (PBL2)
  - Alloy : 130h (PBL2)
  - SPIN : 90~100h (PBL3)
- 工数
  - 935人時(PBL3)

# 作り込み欠陥数と発見欠陥数



※下流工程の欠陥は、ネットワーク制御システムの欠陥も含む

形式手法によって、下流工程での手戻りの抑制などを期待できる

# 今後に向けて

---

- 今回のプロジェクトでは、SDNやネットワーク制御システム開発における特有の観点で形式手法を用いたという面は小さい
  - 現在も別のメンバーでQNET様とのPBLを継続しており、"SDN"ならでは、"ネットワーク制御"ならではの問題点や、要件について、形式手法などで解決できないか？などを模索中

# まとめ

---

- ネットワークの分野でもソフトウェア開発手法が重要になってくるはず...
  - 従来のソフトウェア工学的な手法がこの分野でも有効かもしれない
    - 形式手法やドメイン特化型モデル駆動開発
      - 形式手法にも複数あるから、適材適所で使いたい
      - メリット, デメリットを考える
- この分野ならではの課題を見つけ出し, どう解決していくかは今後の課題

ご清聴ありがとうございました

FUJITSU

  
QITO

Copyright© 2012 富士通九州ネットワークテクノロジーズ株式会社. All Rights Reserved.

Copyright© 2012 九州大学 システム情報科学府 QITO. All Rights Reserved.