特集論文

システム試験の効率化／高精度化技術

川崎将人*
大塚　亮*
後沢　忍*

System Testing Technology for Emulating Production Environment

Masato Kawasaki, Ryo Otsuka, Shinobu Ushirozawa

要　旨

近年，金融や交通などの社会を支える情報システムの障害によって，社会全体に多大な影響を与えることが多くなっている。これらの障害は，旧システムから新システムへの移行時に発生するケースが多いことから，障害の要因はシステムが稼働する環境(＝実環境)と試験環境の差異にあると考えた。そこで，実環境を流れる通信データを試験データとして用いることで，実環境に近い状況を再現できる試験を実現することをねらいとする。

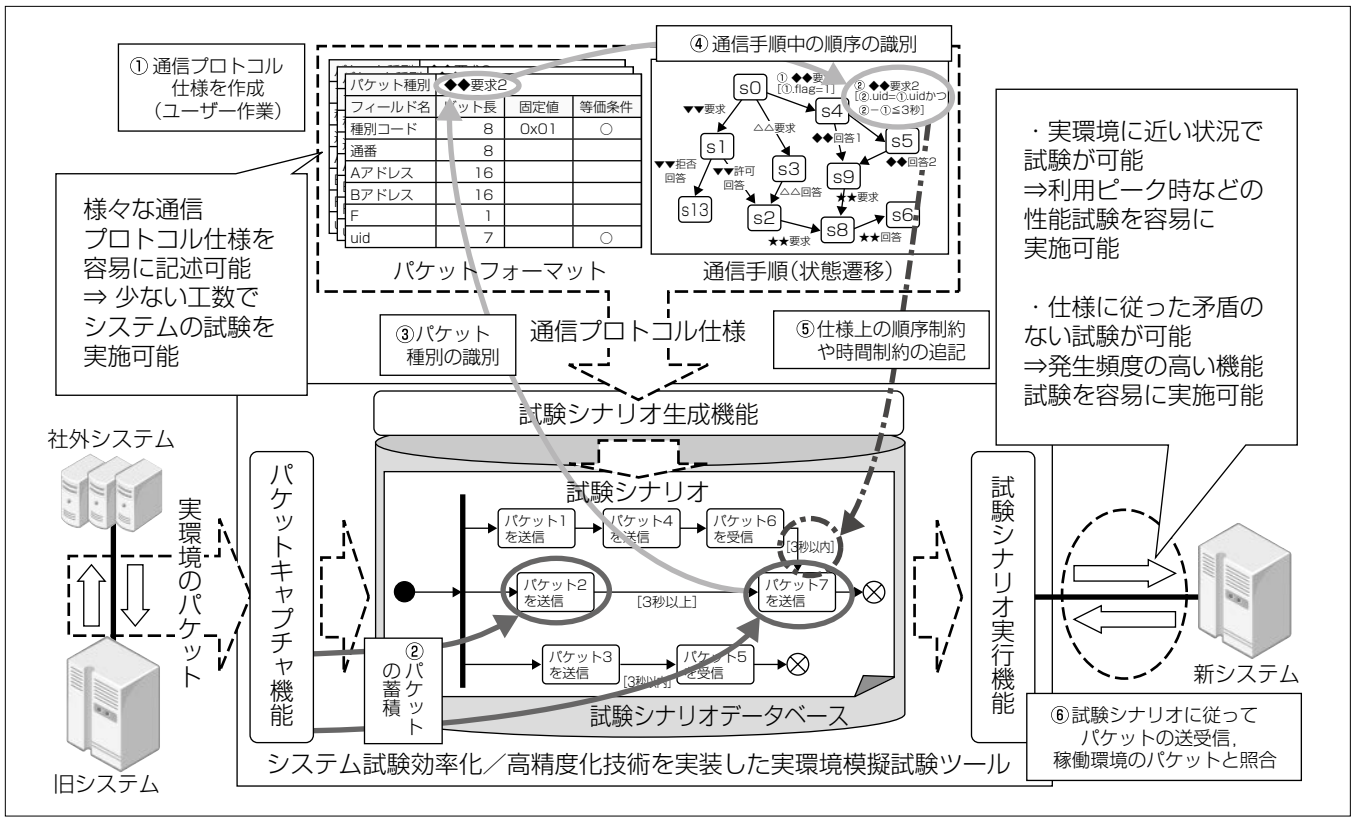
しかし，実環境の通信データを試験環境の新システムに入力しても，出力される通信データは実環境と異なるのが通常である。例えば，送受信の順序やタイミング，パケット数に差異が生じる。その際，試験対象のシステムに不具合があると一概にはいえない。不具合の有無を判別するためには，差異のある通信データが仕様に適合するかを判断する必要がある。そこでこの技術では，仕様への適合性を判別し，通信データの差異を吸収した試験を行えるよう，次の開発を行った。

(1) 仕様上の順序制約や時間制約に従ってパケットの送受信を行うとともに，稼働時と等価なパケットを受信したかを検査する試験シナリオ実行機能

(2) (1)で実行可能な試験シナリオとなるよう，キャプチャしたパケット間に仕様上の順序制約や時間制約を付加する試験シナリオ生成機能

(3) (2)で付加する順序制約や時間制約などの通信プロトコルの仕様の記述に特化した記述形式

この技術によって，少ない工数で様々なシステムの試験が可能になるとともに，実利用に即した機能試験及び性能試験が可能となる。今後は，異常ケースや負荷の追加によって試験精度を更に向上させられるよう，生成した試験シナリオを編集する機能を開発する。



実環境模擬試験ツールの利用手順及び動作概要の説明図

①あらかじめユーザーは，所定の形式に従って，システムが扱う通信プロトコル仕様を記述する。②ユーザーは，既存のパケット キャプチャツールを用いて，実環境の通信データを保存する。試験シナリオ生成機能は，③パケットフォーマットを読み込み，蓄積したパケットの種別を判別し，④通信手順を読み込み，パケットの種別を基にして通信手順中の順序を識別し，⑤識別できた順序と付随する時間制約をパケット間に記録する。⑥試験シナリオ実行機能は，試験シナリオの順序制約と時間制約に従ってパケットの送受信及び実環境のパケットとの照合を行う。

*情報技術総合研究所

1. まえがき

近年，金融や交通などの社会を支える情報システムの障害によって，社会生活に多大な影響を与えることが多くなっている。これらの障害は，旧システムから新システムへの移行時に発生するケースが多いことから，障害の要因はシステムの実環境と試験環境の差異にあると考えた。

一方，システム移行は，長い保守期間中に，ハードウェアなどの寿命によってプラットフォームのみのリプレースが行われることが多い。その際，リプレース後も外部から見て同じ動作を保証する必要がある。つまり，システムの主要なインタフェースであるネットワークで，ネットワークを介してやりとりされる通信データが外部のシステムから見て一致することを保証する必要がある。

そこで，実環境と試験環境の差異を埋めるため，実環境を流れる通信データを利用して試験することを考える。すなわち，実環境を流れる通信データを試験環境の新システムに入力し，出力される通信データが実環境の通信データと等価であるか確認する。これによって，実環境に近い状況を再現し試験することをねらいとする。

しかし実環境の通信データを試験環境の新システムに入力しても，出力される通信データは実環境と同じではないのが通常である。例えば，送受信の順序やタイミング，送受信するパケット数などの差異が発生する。しかしながら，その差異だけで一概にシステムに不具合があるとはいえない。不具合の有無を判別するためには，通信データの差異が仕様に適合するかを判断する必要がある。そこで，通信データの差異が仕様に適合するかを自動的に判断する実環境模擬試験ツールを開発した。

2. システム試験効率化／高精度化技術の実現

2.1 実環境模擬試験ツールの概要

この技術を実装した実環境模擬試験ツールによって，実環境と試験環境の通信データの差異が，仕様に適合するかを自動的に判断することができる。そのためには，プロトコル仕様の順序制約及び時間制約に従って，キャプチャしたパケットを送信し，試験対象から出力されるパケットを受信の上，実環境のパケットと比較し一致しているか検査する必要がある。さらにそれには，試験対象システムが扱う通信プロトコル(＝アプリケーションプロトコル)の仕様を解釈し，キャプチャしたパケット間に仕様上の順序制約及び時間制約を付加する機能が必要である。また，このツールが仕様を解釈できるよう，ユーザーが所定の形式に従って仕様を記述する必要がある。次に，実環境模擬試験ツールの実現方式について述べる。

2.2 実現方式の概要

図1に，実環境模擬試験ツールの機能構成要素を示す。

(1) パケットキャプチャ機能

実環境を流れるパケットをすべて受信し，試験シナリオデータベースに蓄積する機能。Wireshark(1)(注1)などの既存ソフトウェアによって実現されている。

(2) 試験シナリオデータベース

試験シナリオを蓄積する。試験シナリオの構成要素には，送信するパケット，受信を期待するパケット，順序制約情報，時間制約情報などがある。

(3) 通信プロトコル仕様の記述形式

通信プロトコルの仕様記述に特化している。

(4) 試験シナリオ生成機能

(3)の形式で表された仕様を解釈し，試験シナリオデータベースに蓄積されたパケットを解析して，パケット間に仕様上の順序制約情報や時間制約情報を追記する。

(5) 試験シナリオ実行機能

試験シナリオ中の順序制約情報や時間制約情報に従ってパケットの送受信を行うとともに，稼働時と等価なパケットを受信したかを検査する。

次に，特に重要な(3)(4)(5)について詳細に述べる。

(注1) Wiresharkは，Combs, Gerald C.の登録商標である。

2.3 通信プロトコル仕様記述形式

通信プロトコルの仕様には，パケットフォーマットと通信手順の二つがある。

2.3.1 パケットフォーマット

パケットフォーマットの入力イメージを**図2**に示す。

パケットフォーマットは，TCP(Transmission Control

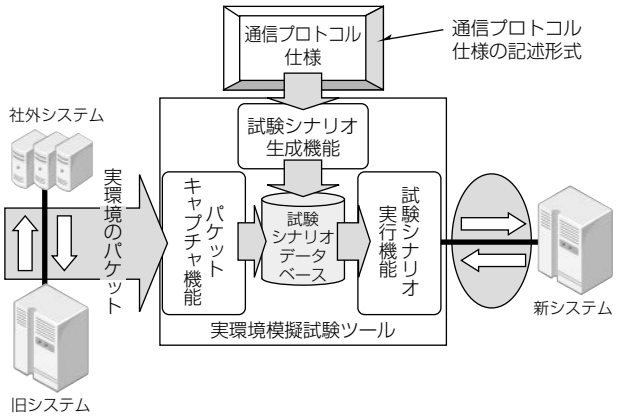


図1．実環境模擬試験ツールの機能構成要素

パケット種別	◆◆要求2		
フィールド名	ビット長	固定値	等価条件
種別コード	8	0x01	○
通番	8		
Aアドレス	16		
Bアドレス	16		
F	1		
uid	7		○

図2．パケットフォーマットの入力イメージ

Protocol)などの下位プロトコルのパケットのペイロードを基にして，アプリケーションプロトコルのパケットのフィールド値を解析するための仕様を記述する。**図2**に示すように，パケット種別ごとに表を持ち，表は少なくとも，フィールド名，ビット長，固定値，等価条件の列を持つ。パケット種別はパケットが持つフィールドの集合を識別するためのものである。フィールド名は，パケットが持つフィールドの名前を示す。ビット長は，フィールド値を格納するのに必要なビット数を示す。固定値は，ビットパターンで表されたパケットから，その種別を識別する際に使用するもので，特定のパケット種別に該当するためには，ビット長によってパケットを分割後にそのフィールド値が所定の値になることを定義する。等価条件は，実環境と試験環境のパケットを比較する際に使用するフィールドを指定する。

2.3.2 通信手順

通信手順の入力イメージを**図3**に示す。

通信手順は，送受信されたパケット間の依存関係を解析し，パケット間に順序制約と時間制約を記録するために使用する。アプリケーションプロトコルと1対1で定義される。

通信手順の基本構造は状態遷移モデルであり，特にプロトコル状態マシンと呼ばれるモデルを採用している。プロトコル状態マシンは通信路における状態遷移を表しており，TCPの仕様記述にも使用されている[2]。

プロトコル状態マシンにおける“状態”とは，通信路から観測可能な状態を表す。例えば，“パケットAを送信後”のような状態を表す。プロトコル状態マシンにおける“遷移”とは，パケットの送受信イベントによる状態の変化を表す。例えば，“パケットAを送信前”から“パケットAの送信”イベントによって“パケットAの送信後”へと遷移する。

プロトコル状態マシンの形式は，OMG(Object Management Group)が規定するUML(Unified Modeling Language)(注2)のように標準化された形式や，Alfaro[3]らのように独自に規定した形式もある。この技術では，一般的な状態遷移モデルに，次の二つの記述制約を追加した形式を使用する。一つ目の制約として，遷移を引き起こすためのイベントには，パケットフォーマットで規定したパケット種別を指定する。二つ目の制約として，状態遷移する際の条件を表すガード条件に対し，次の3種類の組合せという制約を課す。

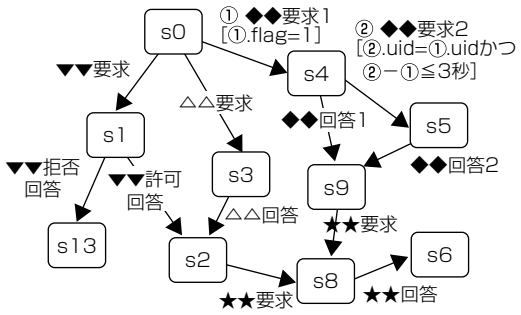


図3．通信手順の入力イメージ

(1) 送受信するパケットのフィールド値と，所定の定数との不等式。例：**図3**中，遷移①の(①.flag=1)。

(2) 送受信するパケットのフィールド値どうしの不等式。例：**図3**中，遷移②の(②.uid=①.uid)。

(3) 送受信するパケットの送受信時刻どうしの演算結果，及び時間差を表す定数との比較演算による不等式。この制約は，時間制約を表している。
例：**図3**中，遷移②の(②−①≦3秒)。

複数の制約が組み合わされている場合には，それらの制約をすべて満たすこと，すなわち論理積を表すものとする。

(注2) UMLは，Object Management Group Inc.の登録商標である。

2.4 試験シナリオ生成実現方式

試験シナリオ生成機能の概要を**図4**に示す。

試験シナリオ生成は，①パケット種別の解析，②パケット間の依存関係の解析，③順序制約と時間制約の追記の，三つの手順で行う。このうち②が試験シナリオ生成処理の要となる処理である。

2.4.1 パケット種別の解析

パケット種別解析処理の概要を**図5**に示す。

パケット種別の解析処理は，蓄積したパケットが，どのパケット種別に該当し，どのようなフィールドと値を持つかを識別するための処理である。処理のポイントを次に示す。

(1) パケットをフィールドに過不足なく分割できるか？

(2) 分割したフィールドの値が，固定値カラムの値で示した値すべてに一致するか？

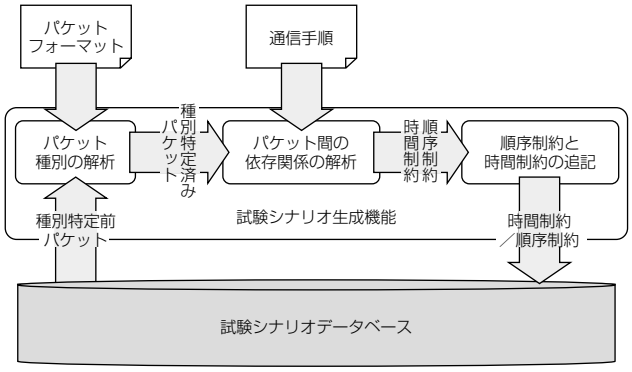


図4．試験シナリオ生成機能の概要

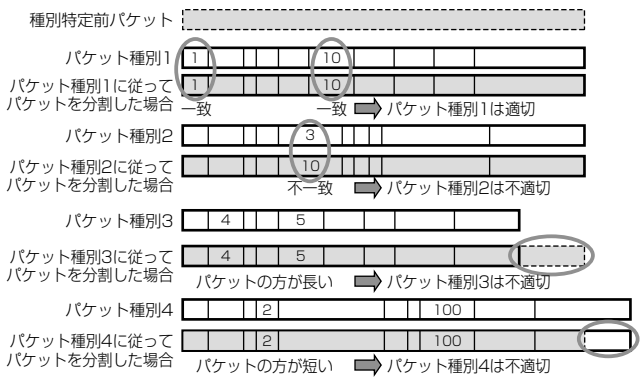


図5．パケット種別解析処理の概要

なお，可変長のパケットに対応する場合や，下位プロトコルのパケット複数個からアプリケーションプロトコルのパケットを得る場合でも，この技術を拡張することで容易に対応することができる。

2.4.2 パケット間の依存関係の解析

パケット間の依存関係解析処理と制約追記処理の概要を**図6**に示す。

TCPで接続の確立から切断までがそうであるように，一つの送受信シーケンスは，通信手順における初期状態からの遷移の系列(＝状態遷移パス)に相当する。そこで，一つの状態遷移パスを表すデータとしてトークンを使用する。**図6**に示すように，トークンは記録したパケットと，通信手順における遷移の対応付けのリストを持つ。

このような前提の下，パケット間の依存関係を解析していく。依存関係を解析する上での基本動作は，次々にパケットを取得し，次の2点を解析する処理である。

(1) 取得したパケットがどのトークンに属するか？(**図6**中でトークンと対応付けの間の実線に相当)
(2) 取得したパケットが通信手順上のどの遷移に該当するか？(**図6**中の破線矢印に相当)

なお，状態遷移の終了状態に到達した場合には，パケットをトークンに対応付ける対象とはしない。これはTCPにおけるコネクションの切断に相当する。また，取得したパケットが，状態遷移の最初の遷移を起こすイベントと一致する場合には，トークンを新たに生成する処理を行う。これはTCPにおけるコネクション確立処理の開始に相当する。

2.4.3 順序制約と時間制約の追記

パケット間の依存関係を解析後，識別された順序制約とそれに付随する時間制約を試験シナリオに追記する。**図6**では，処理①～④の実線矢印で示した処理が該当する。

図6の処理①は，順序制約の追記処理である。トークンが保持する隣り合う対応付け(**図6**中の対応付け①②)がそれぞれ指すパケット(**図6**中のパケット①②)の間に，順序制約情報(**図6**中の順序制約①)を追記する。

ガード条件の追記処理のポイント3点を，**図6**を用いて次に述べる。それぞれ**2.3.2項**の(1)～(3)に対応する。

(1) 定数値との比較の場合は，フィールド値制約情報(フィールド値制約①③)として試験シナリオに追記(処理②)。
(2) フィールド値同士の比較の場合（例えば遷移②の条件“②.uid=①.uid”)は，先行する他の遷移(遷移①)に対応するパケット(パケット①)の具体的なフィールド値（uid=5と仮定する)に変換し，フィールド値制約情報（フィールド値制約②)として試験シナリオに追記(処理③)。
(3) 時間制約の場合は，対応するパケット同士(パケット①②)の間に時間制約情報(時間制約①)を追記(処理④)。

2.5 試験シナリオ実行実現方式

生成された試験シナリオは，UMLアクティビティ図に

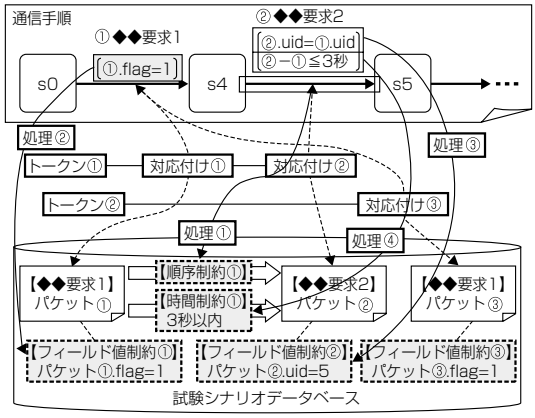


図6．依存関係解析処理と制約追記処理の概要

似たデータ構造を持っている。すなわち，蓄積されたパケットがアクション，順序制約が制御フロー，時間制約がガード条件に対応する。そのため，紙面の都合上，説明は省略するが，**2．4節**の結果出力される試験シナリオを基に，UMLアクティビティ図のデータ構造に容易に変換できる。

したがって，試験シナリオ実行時の基本動作は，UMLアクティビティ図のデータ構造で，開始ノードから終了ノードに至るまでのパスをたどる動作になる。さらにこの技術では，次の二つを行う。

一つ目は，パケットを送受信した際に，その時刻を記録する。これは，後に送受信するパケットの送受信時刻が時間制約を満たすか検査するためである。

二つ目は，受信パケットのアクションでは，パケットの受信を待機するだけでなく，パケットを受信した際には試験シナリオ中のパケットと比較するとともに，時間制約を満たすか検査する。比較時には，**2.3.1項**で示したパケットフォーマットで，等価条件カラム値に“○”を記述したフィールドを対象として比較を行う。対象となるフィールドすべてについてフィールド値が一致した場合に限り，パケットは等価であると判断する。

3. むすび

この技術によって，少ない工数で様々なシステムの試験が可能になるとともに，実利用に即した機能試験及び性能試験が可能となる。今後は，異常ケースの追加，負荷の追加によって試験精度を更に向上させられるよう，生成した試験シナリオを編集する機能を開発する。

参考文献

(1) Wireshark：http://www.wireshark.org/
(2) RFC 793-Transmission Control Protocol：http://www.faqs.org/rfcs/rfc793.html
(3) Alfaro，L. de，et al.：Inteface automata，ACM SIGSOFT Software Engineering Notes，26，Issue 5，109～120（2001）