Requirements Specifications Interaction Analysis

Bill Mitchell
Department of Computing
University of Surrey

Reality of Engineering Requirements Specifications

- Rapid development of lightweight specifications
- Specification by example scenarios
- Poor coverage of scenarios
- Disjoint engineering groups with little communication
- Extreme pressure to ship on time

Hence large number of defects only found in field testing

Research Project Pilot

FATCAT an internal tool, takes sets of scenarios (in e.g MSC notation):

- analyse scenarios coverage
- find conflicts
- construct interesting FI tests

Current Pilot coordinated by Motorola UK Research lab with:

- UI Design Group
- UI Software Tool Development Labs
- System Architecture Group
- Testing Group
- Testing Tools Software Management Group
- Requirements Specification Managers

- EU 6th Framework STReP, Forming a consortium:

Standard Techniques

Internal study of 200 TETRA MSC scenarios with Spin and standard MSC automata synthesis algorithms (Alur, Leue etc).
Result:

- state explosion
- generated many “bogus” errors

Two other Motorola Research Labs built POTS system with Switch in SDL for model checking with FDR.
Result:

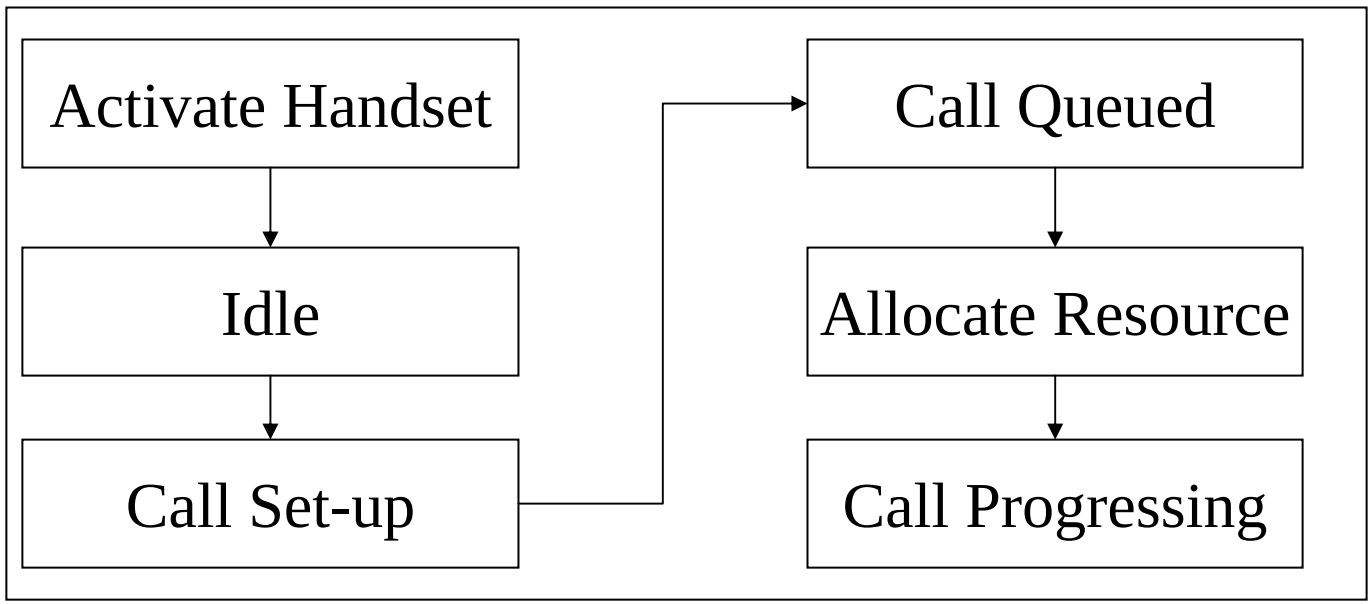
- state explosion problems
- only found conflicts when the algorithm directed towards error states

For wireless telecomms specifications, standard model checking not always the right approach

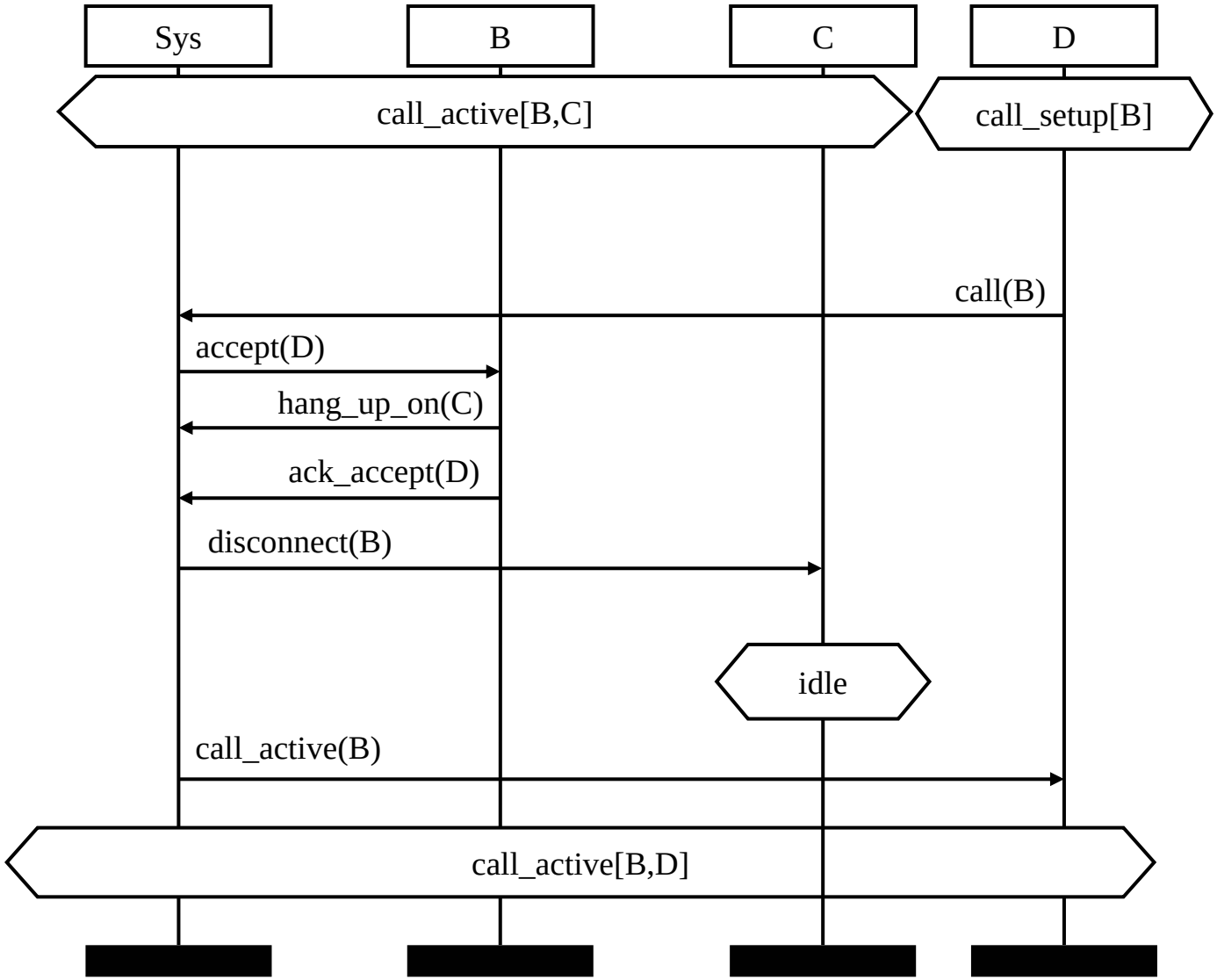
Purpose of MSC Specifications

Specs define Phase Transitions

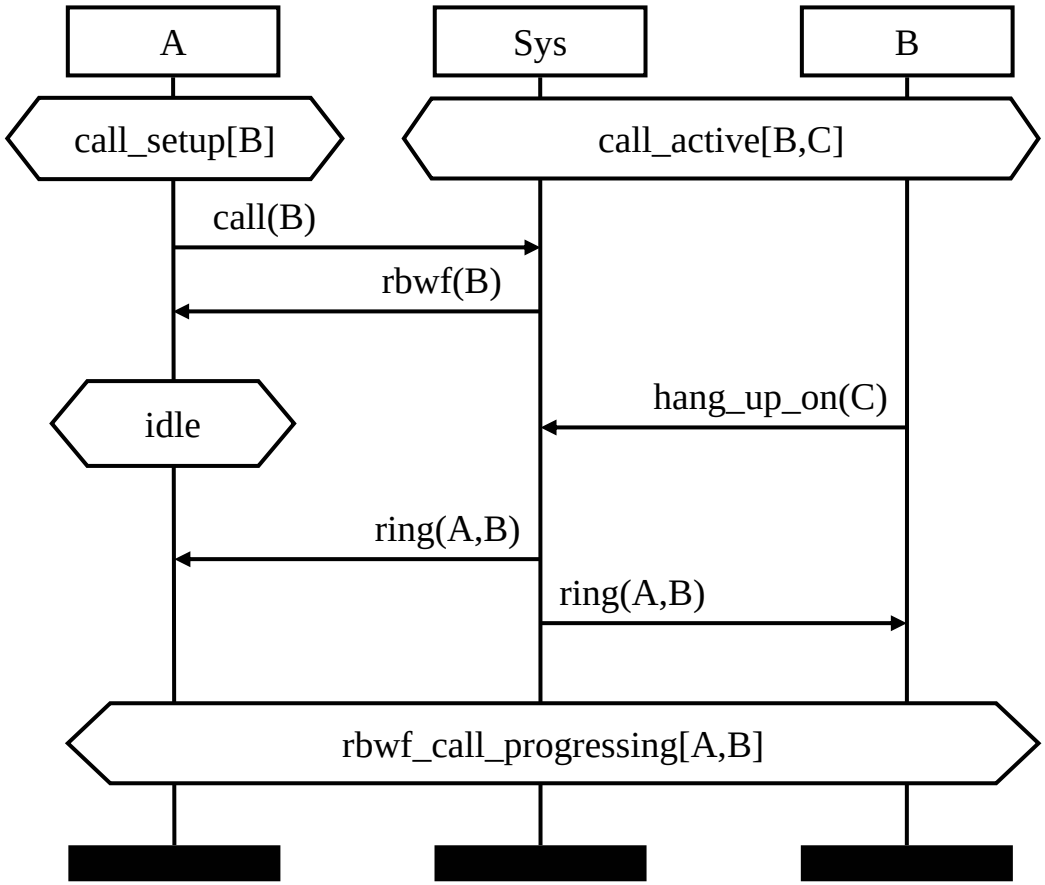
- Each phase describes a meaningful part of the overall operation of the system.
- Each phase involves its own mini-protocol.
- Each MSC shows how to move from one phase to another, via intermediate phases.



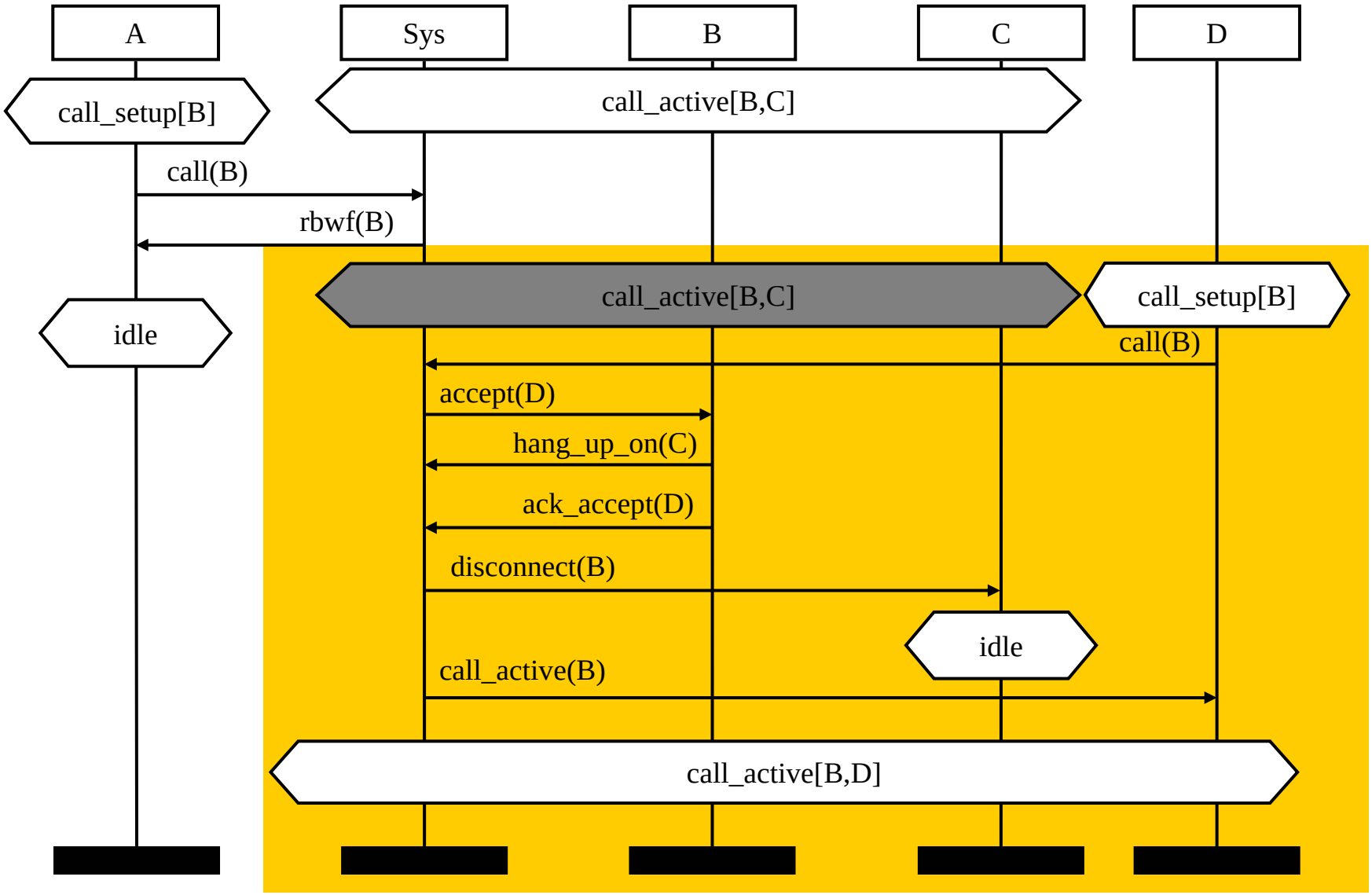
Example, Call Waiting from paper in FIW 2000



Example, Ring Back When Free, from paper in FIW 2000



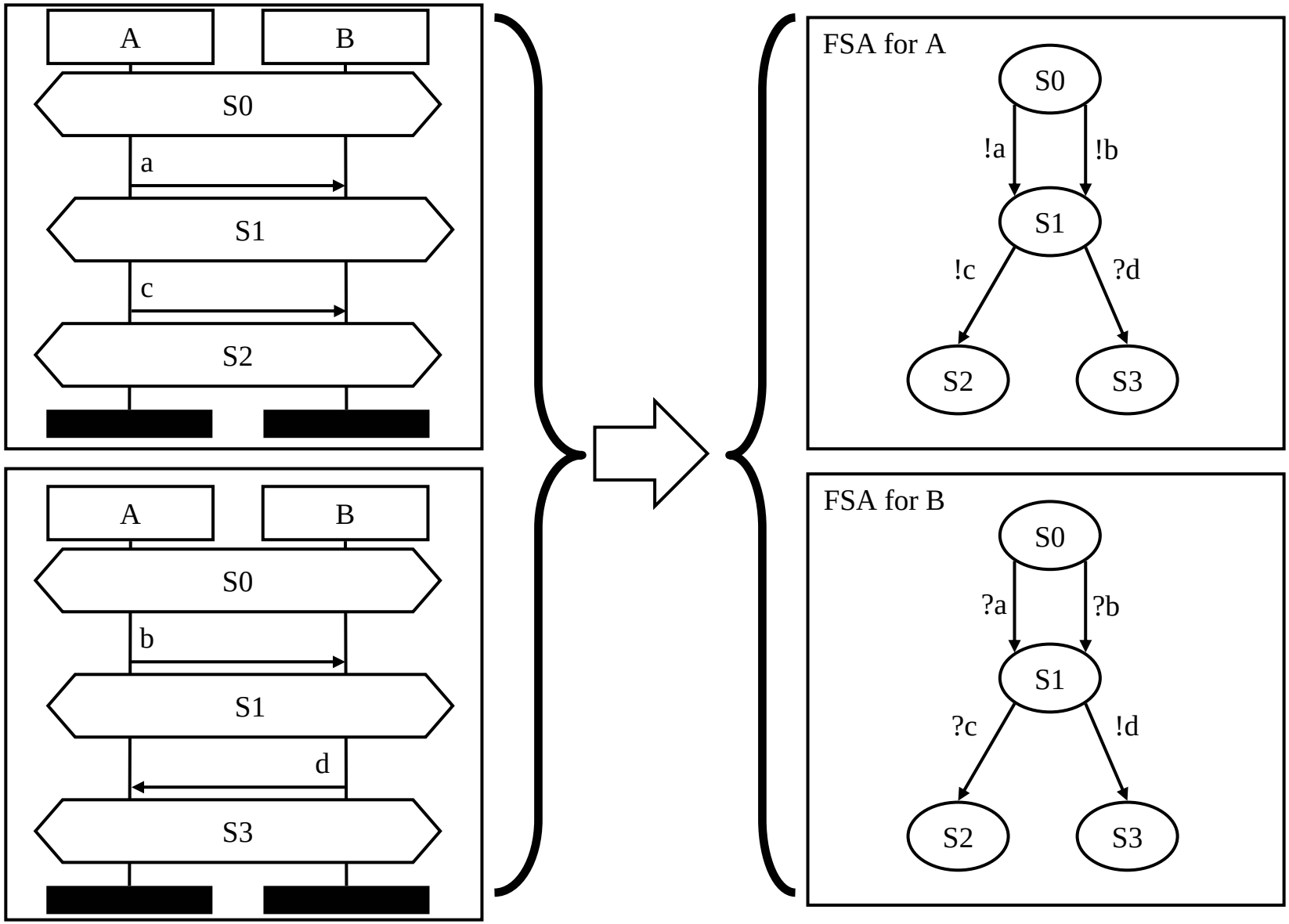
Example, FI from paper in FIW 2000



Naive Deterministic FSA implementation

- Each instance to be implemented as a FSA
- Next state after each event is unique except where this causes nondeterminism
- FSA must be deterministic
- Phases define unique states in implementation

Deadlock Example

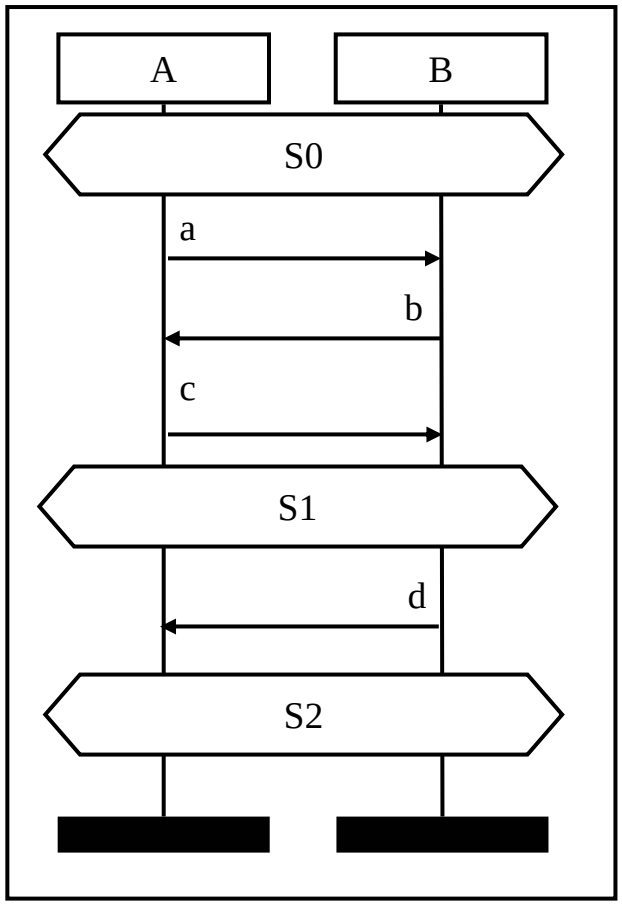


Deterministic FSA implementation with Phases

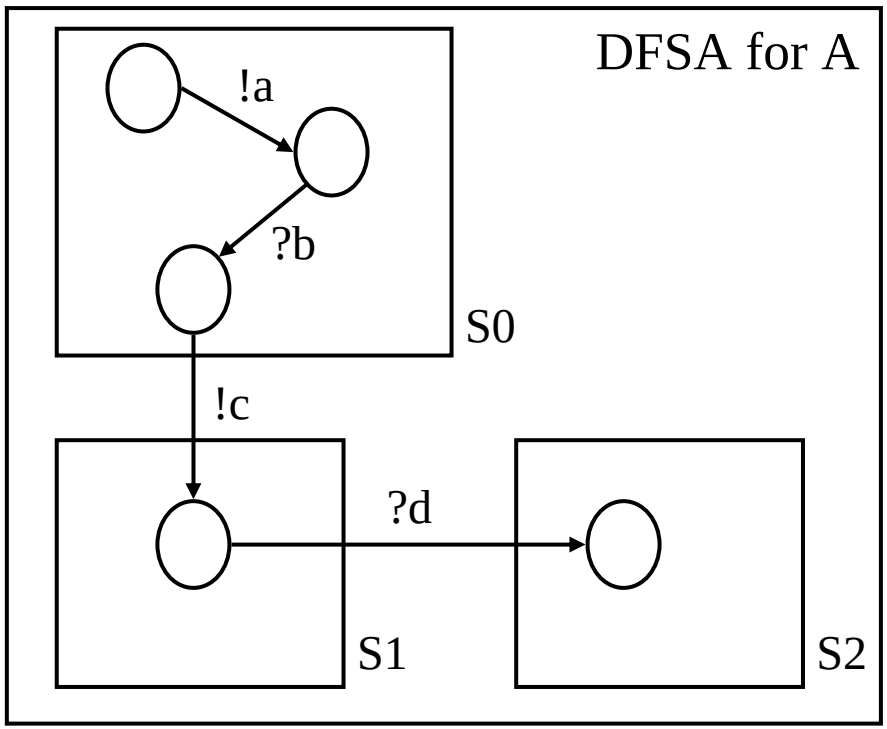
Better Semantics?

- Each instance to be implemented as a DFSA
- Phases correspond to set of states in DFSA (composite state)
- Restrict construction of DFSA from MSC via additional semantics of phases.

Phase Traces, Phase Automaton



Phase trace for A:
S0, !a, S0, ?b, S0, !c, S1, ?d, S2



Additional Phase Semantics, First Attempt

Booby Trapped Semantics

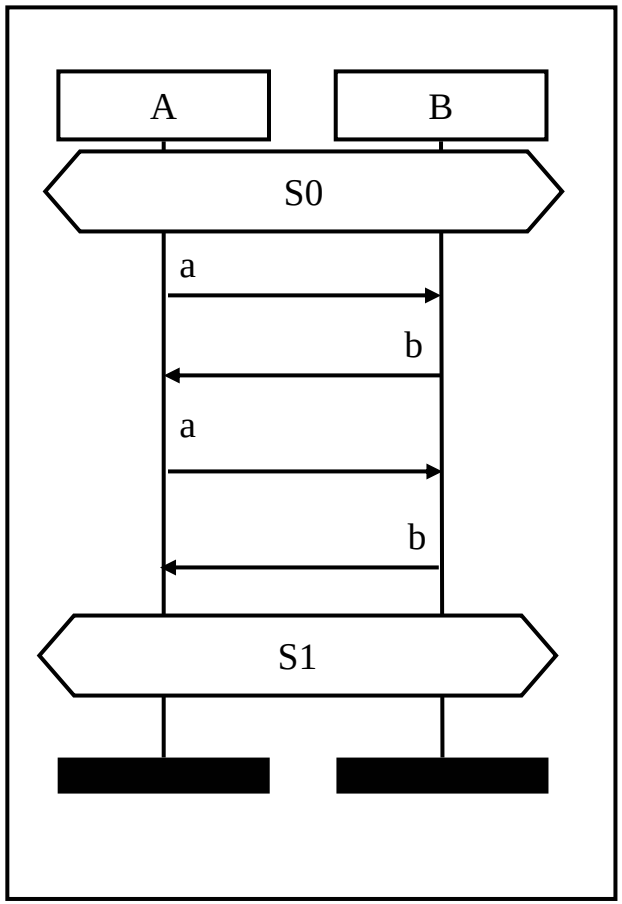
If

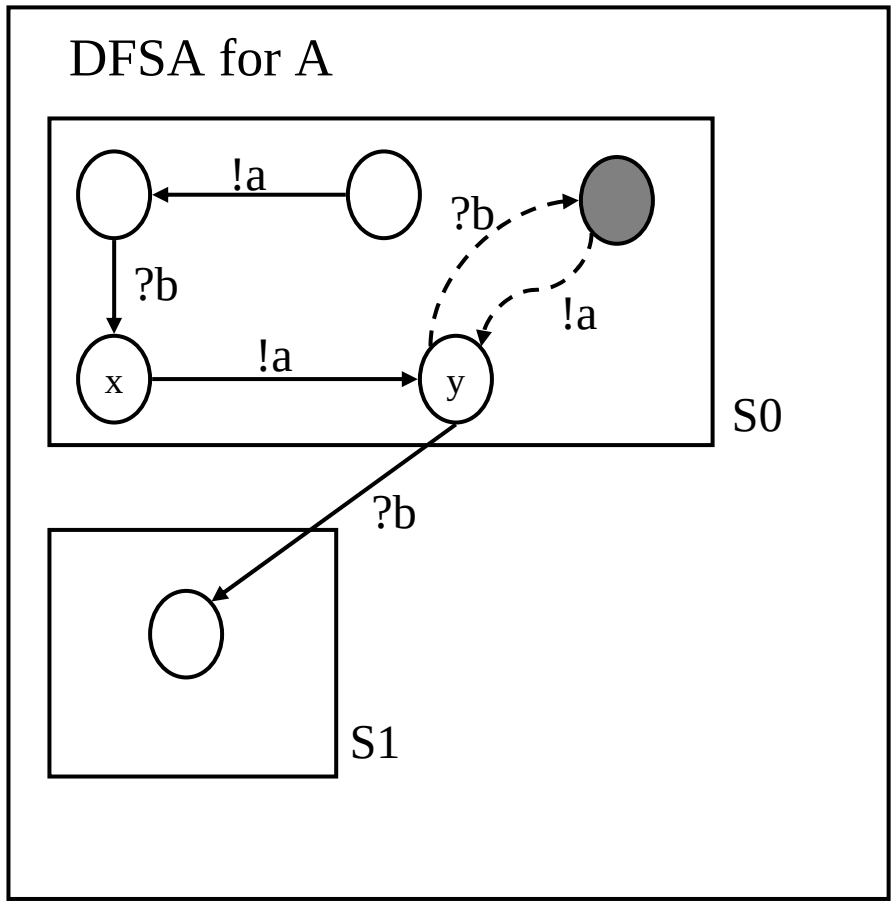
a DFSA path has an execution trace which matches the initial section of an MSC phase trace

then

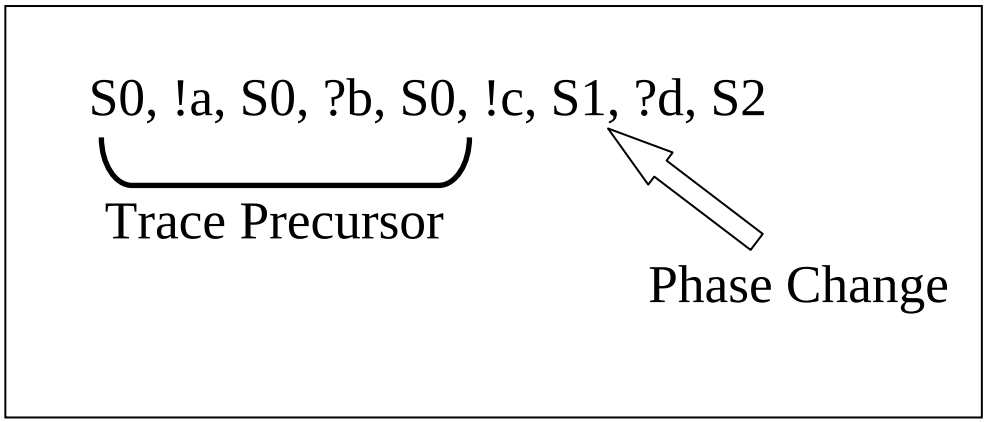
the implementation can always generate the rest of the phase trace from that point.

Consequences of Bad Semantics





Corrected Additional Phase Semantics



If

a DFSA path has an execution trace that matches a trace precursor

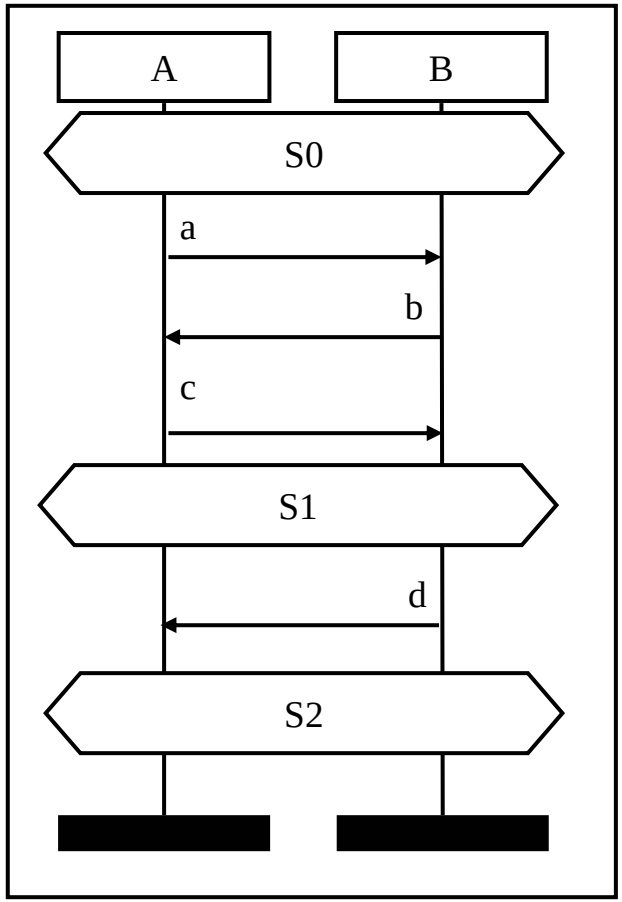
and

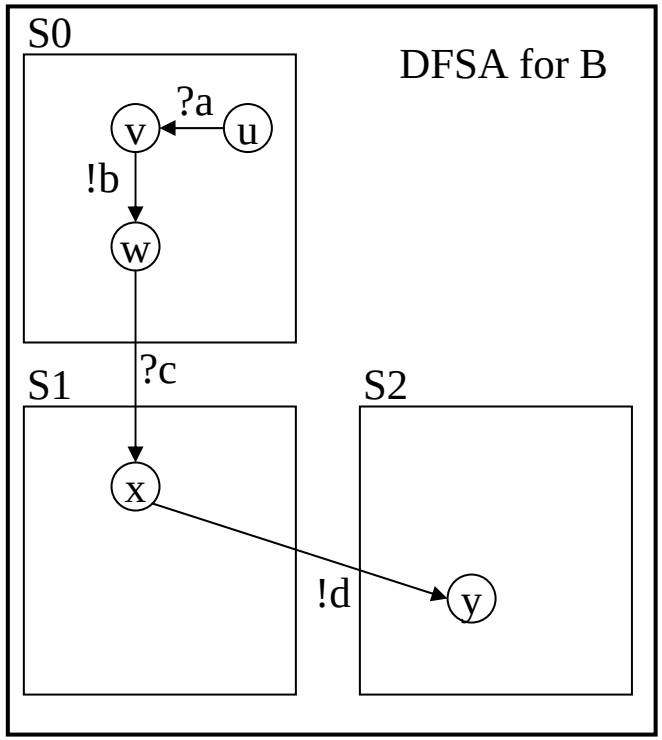
it has reached a point where a phase transition is possible

then

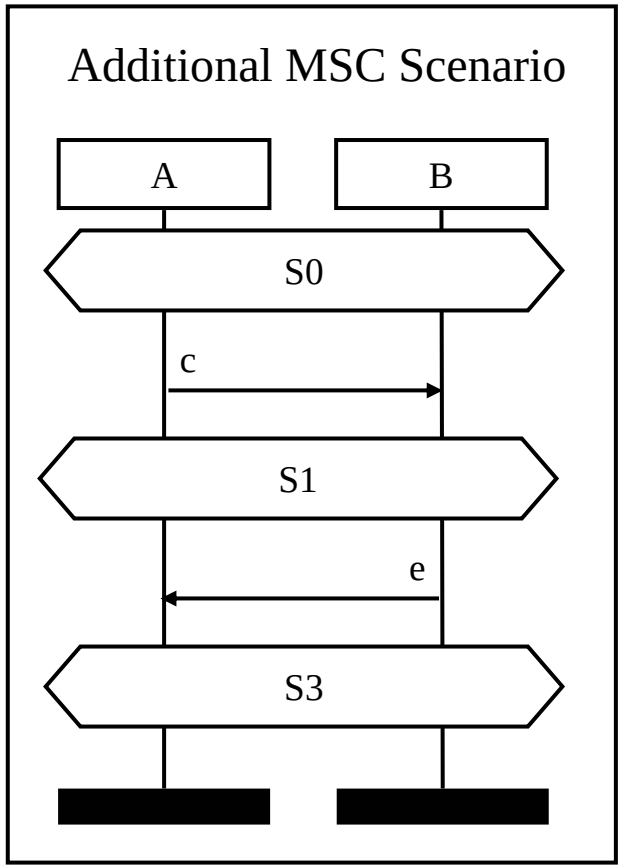
the implementation can always generate the rest of the phase trace from that point.

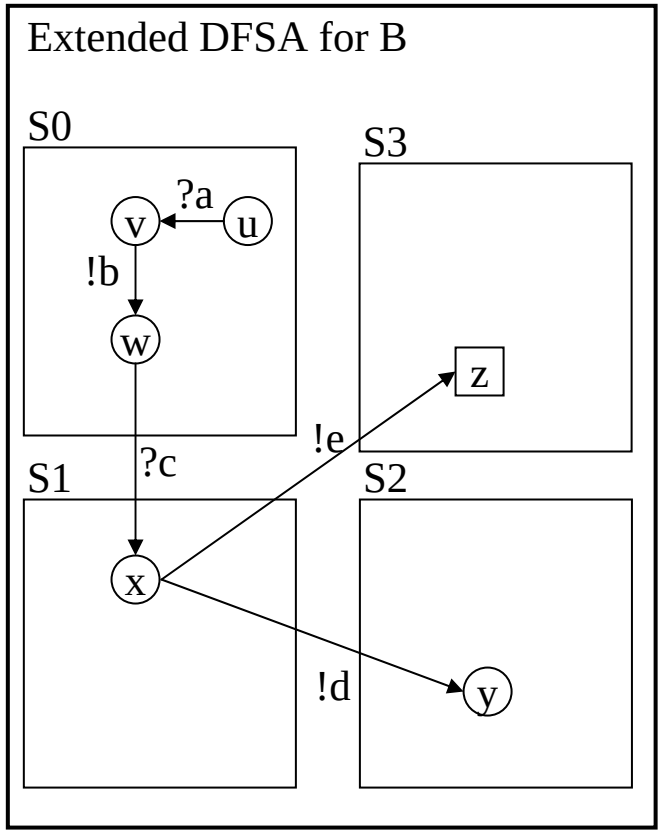
Overlapping Threads





Overlapping Threads



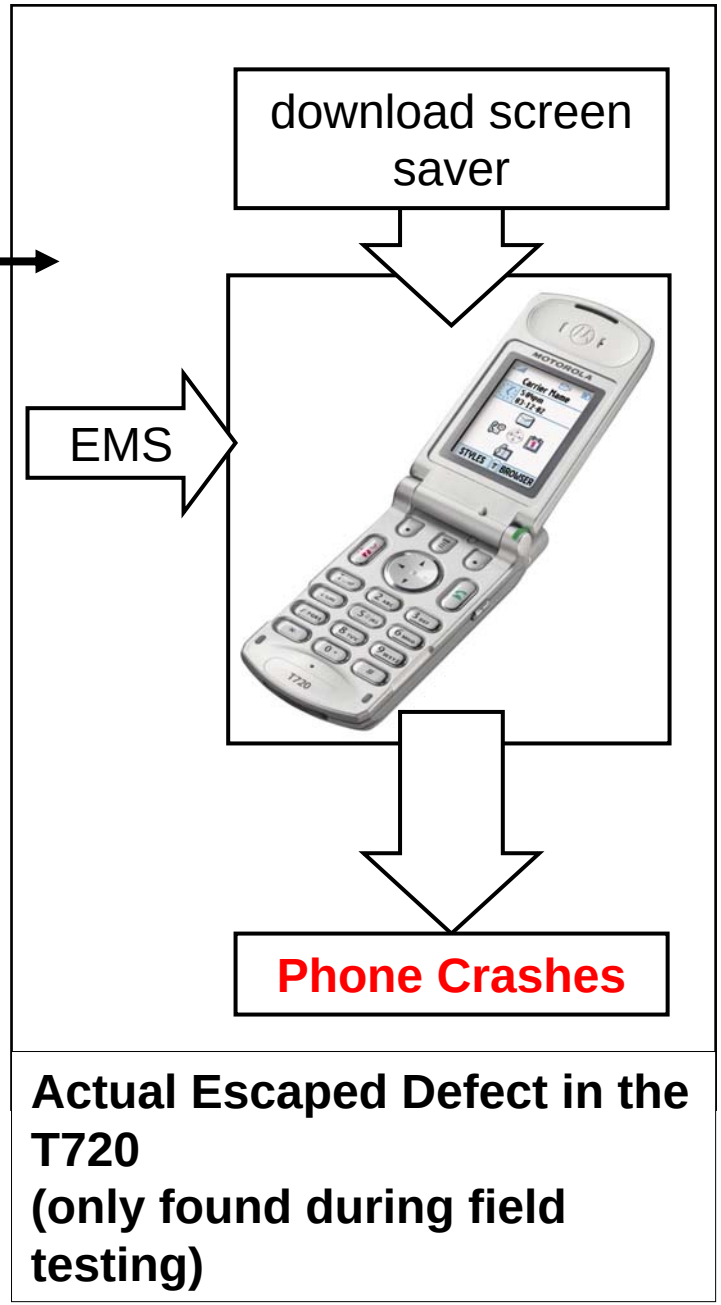


Missing Scenarios for Browser Suspend MRS

Missing Scenarios

Scenario 1: browser active downloading file, incoming EMS

Scenario 2: browser suspended, voice call active, incoming EMS



Actual Escaped Defect in the T720
(only found during field testing)

Could generate 14 missing use cases.

FATCAT only outputs 2.
These are the significant ones.

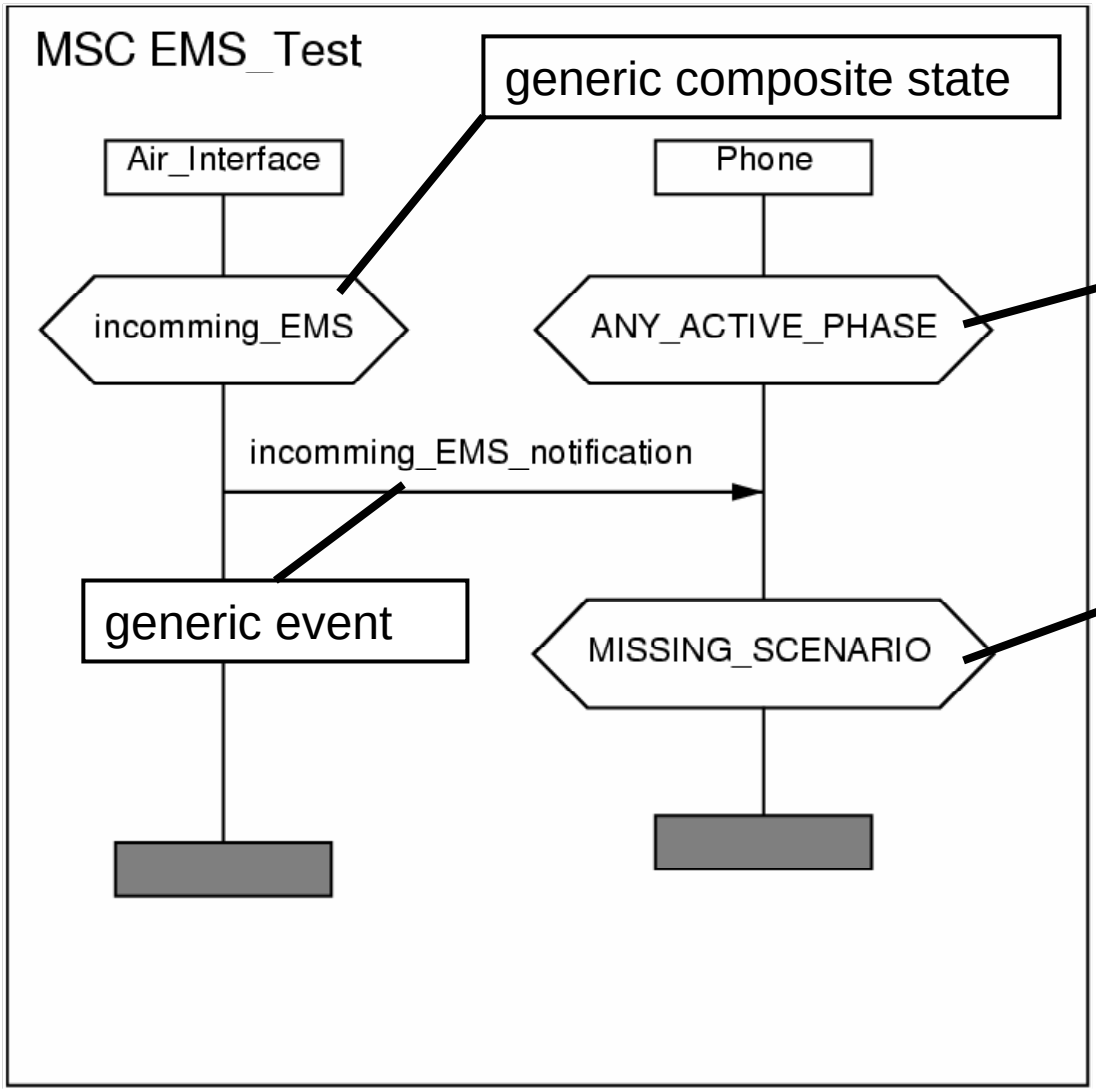
Missing scenarios could be used purely for test purposes, or could be used for additional requirements specifications.

Process Algebra for overlapping recursive processes

par	$P \mid Q$	$=$	$P \lhd Q + P \rhd Q$
skew1	$a \cdot P \lhd b \cdot Q$	$=$	$a \cdot P \lhd\mid b \cdot Q$ if $a \sqsupset b$
skew2	$a \cdot P \lhd b \cdot Q$	$=$	$a \cdot (P \lhd b \cdot Q)$ if $a \not\sqsupset b$
skew3	$P \rhd Q$	$=$	$Q \lhd P$
zero1	$0 \lhd Q$	$=$	0
left-synch1	$a \cdot P \lhd\mid b \cdot Q$	$=$	$a \cdot (P \lhd\mid Q)$ if $a \sqsupset b$ and $\neg\eta(a)$
left-synch2	$a \cdot P \lhd\mid b \cdot Q$	$=$	$a \cdot (P \parallel Q)$ if $a \sqsupset b$ and $\eta(a)$
branch1	$a \cdot P \lhd\mid b \cdot Q$	$=$	$a \cdot P + b \cdot Q$ if $a \not\sqsupset b$ and $\eta(a)$
prune	$a \cdot P \lhd\mid b \cdot Q$	$=$	$a \cdot P$ if $a \not\sqsupset b$ and $\neg\eta(a)$
zero2	$0 \lhd\mid Q$	$=$	0
synch1	$a \cdot P \parallel b \cdot Q$	$=$	$a \cdot (P \parallel Q)$ if $a \sqsupset b$
synch2	$P \parallel Q$	$=$	$Q \parallel P$
zero3	$0 \parallel Q$	$=$	Q
branch2	$a \cdot P \parallel b \cdot Q$	$=$	$a \cdot P + b \cdot Q$ if $a \not\sqsupset b$ and $b \not\sqsupset a$

Questions

Missing Scenarios



Wild card state, matches any composite state that interfaces with Air_Interface process. Since ‘browser_active’ has use case where it accepts ‘?incoming_call’ from Air_Interface, wild card can be identified with ‘browser_active’.

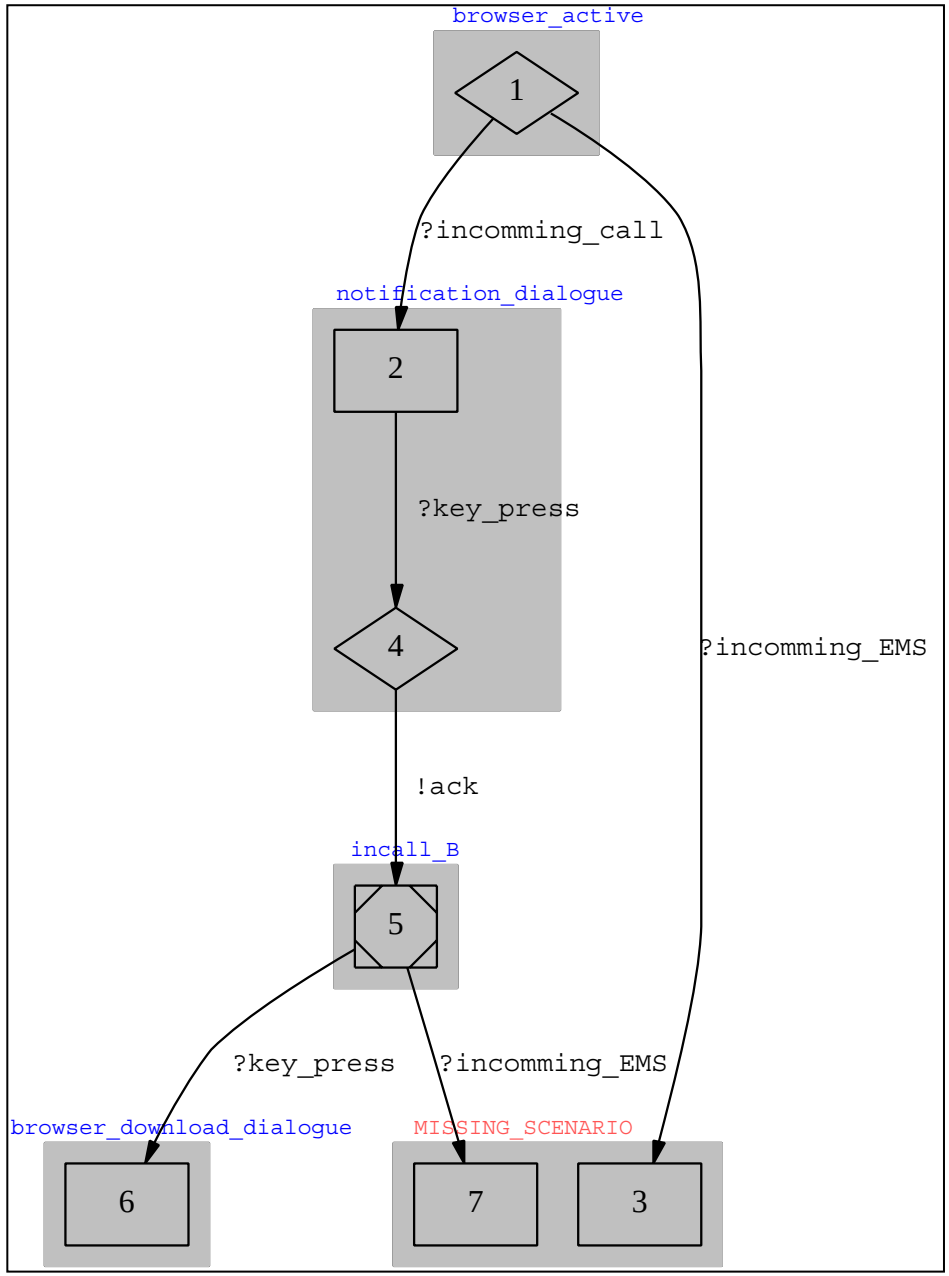
If a trace in the phase automaton can reach here need to check whether this represents a true missing scenario. This will be true exactly where there is not other transition triggered by generic event.

Automatically generate generic abstract use case from UI guide

Phase Automaton for Browser MSC-s

Because ‘Any_Active_Phase’ can match against ‘incall_view’ the phase semantics then force state 5 to transition to ‘MISSING_SCENARIO’ via the ‘?incoming_EMS’ event.

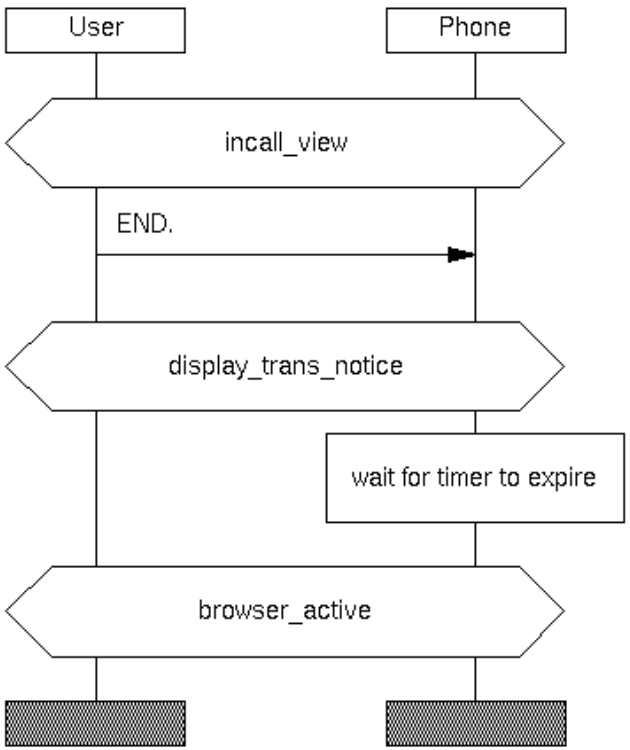
Checking against known use cases shows there is no requirement specification to determine what the correct transition for 5 should be at this point, so this transition does define a missing scenario.

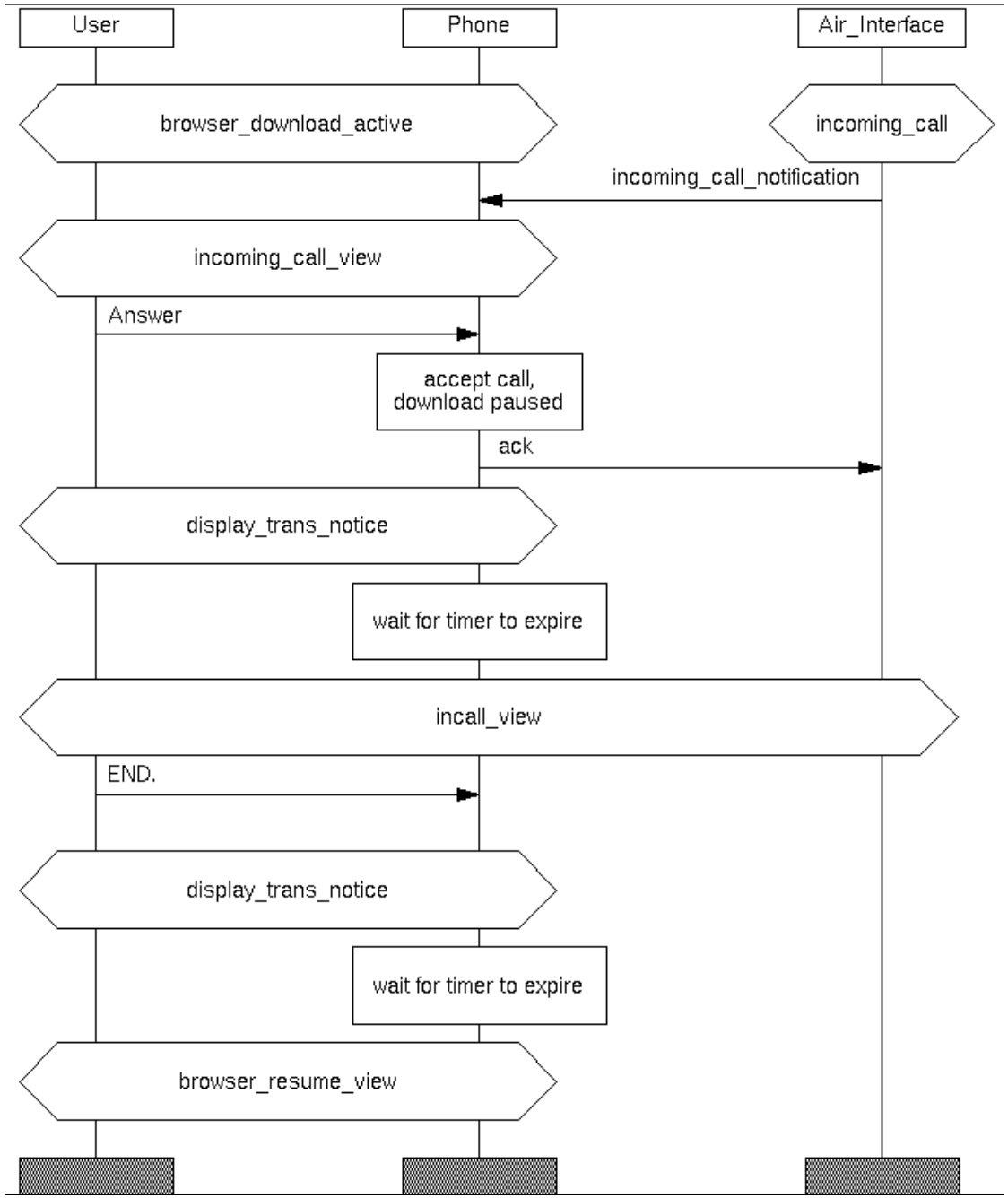


Questions?

MSC Representation of UI Use Cases for WAP Browser

MSC-s based on translation of UI semantics into abstract representation

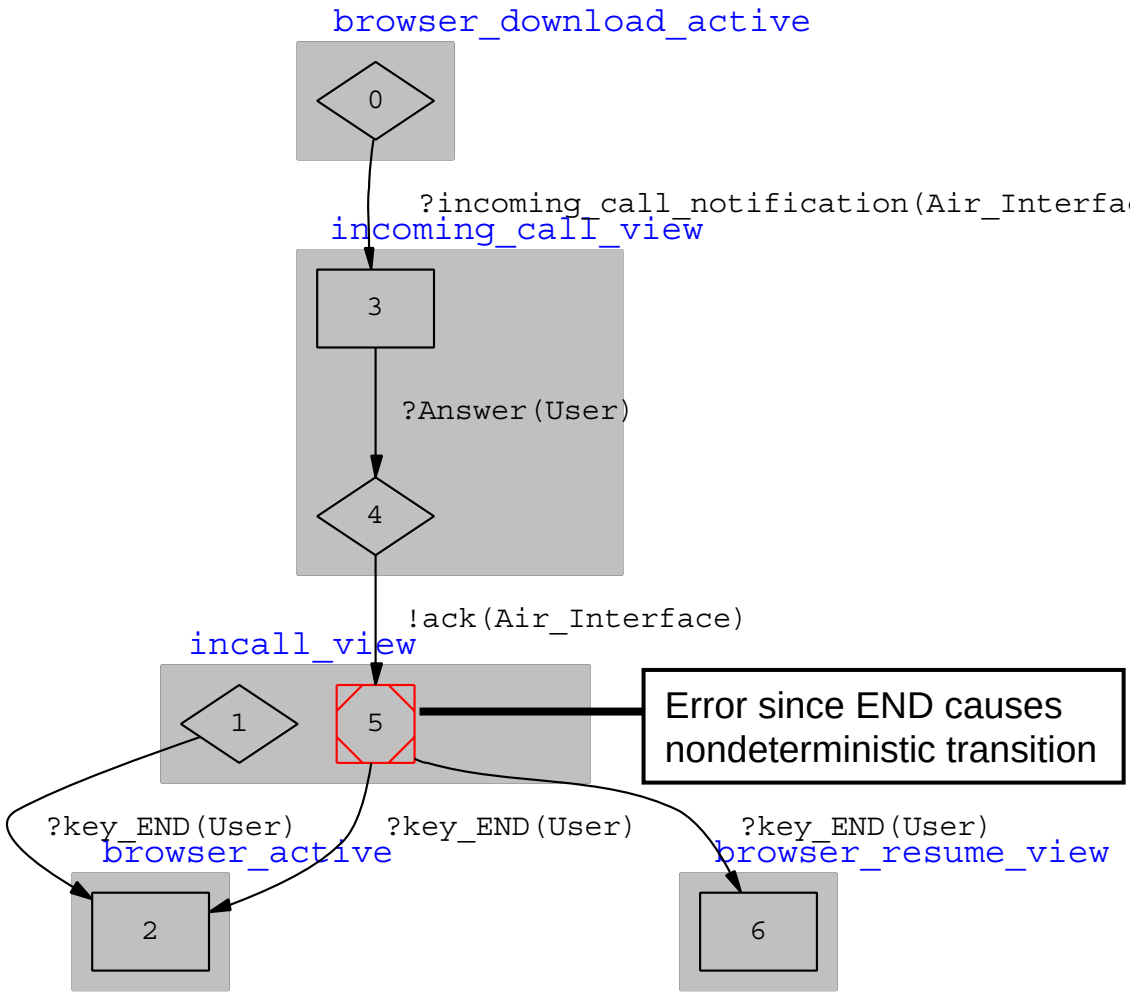




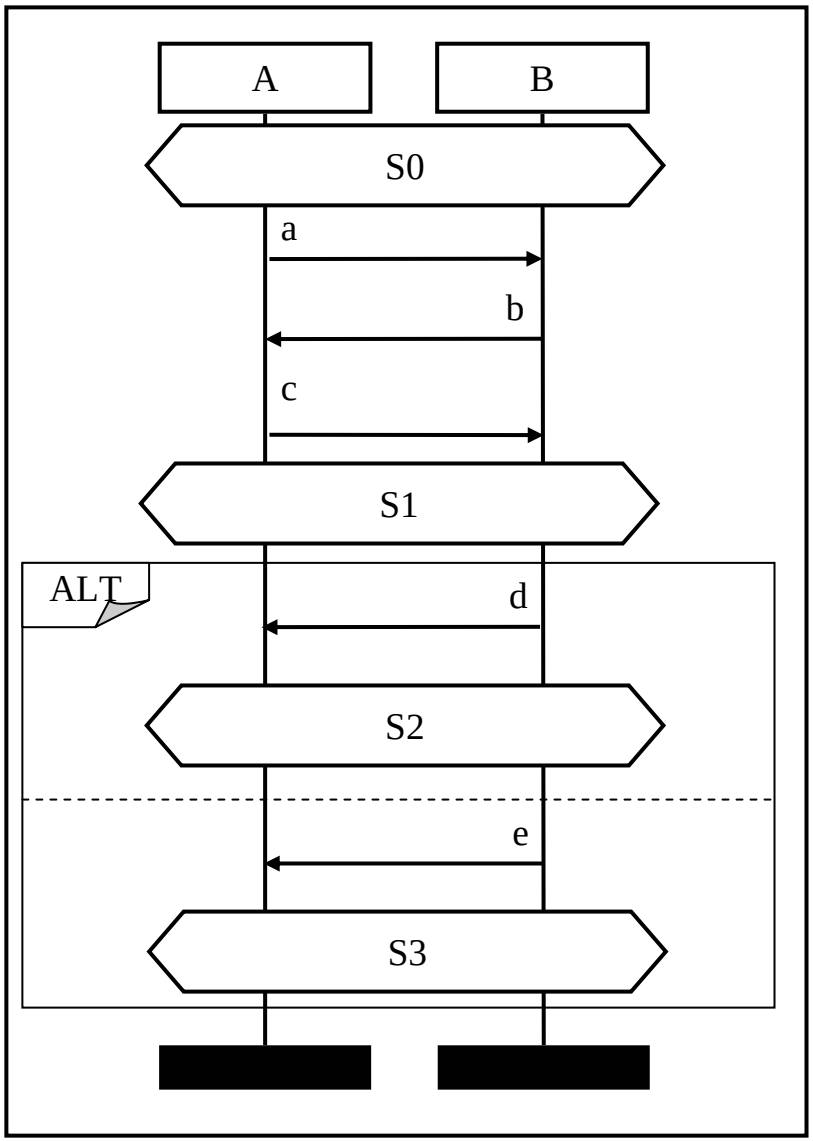
Phase Automaton with Conflict

The phase semantics force the use cases to be combined as shown.

This trivial example does not require the subtlety of the phase semantics, but it clearly illustrates how use cases are combined within a single process representation.

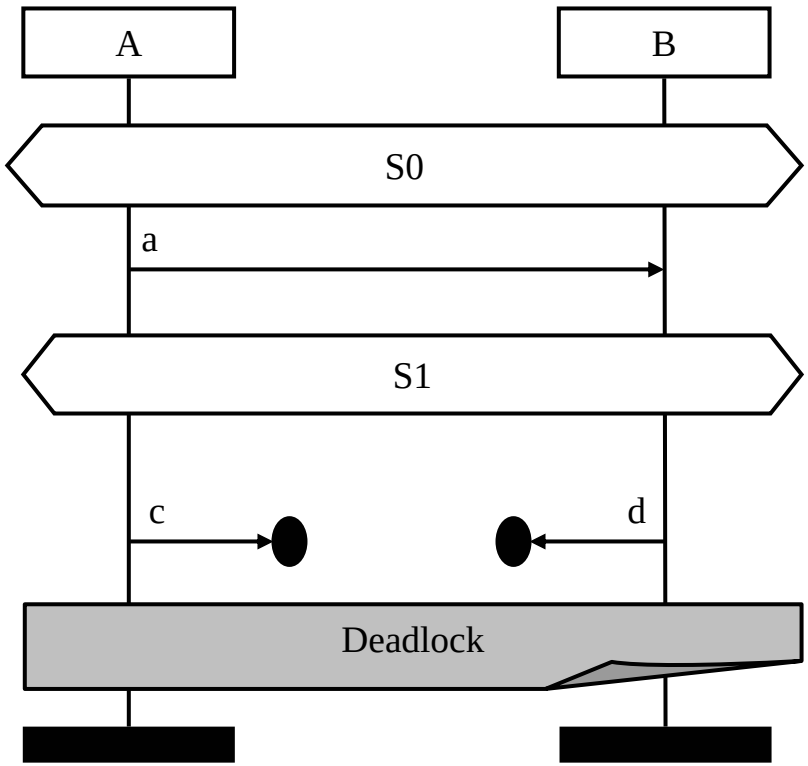


Overlapping MSC

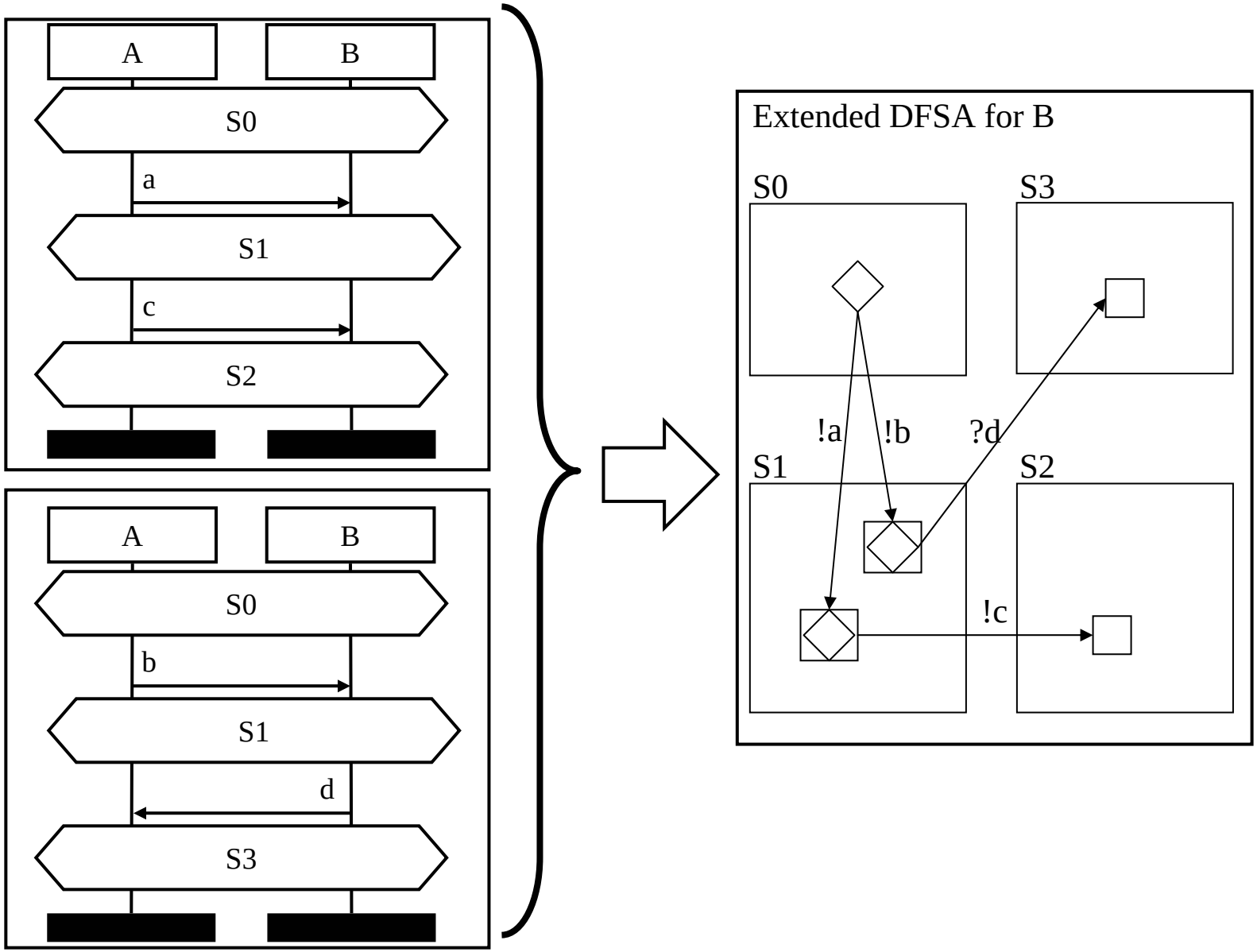


These overlapping MSC can be generated directly from the original MSC-s.

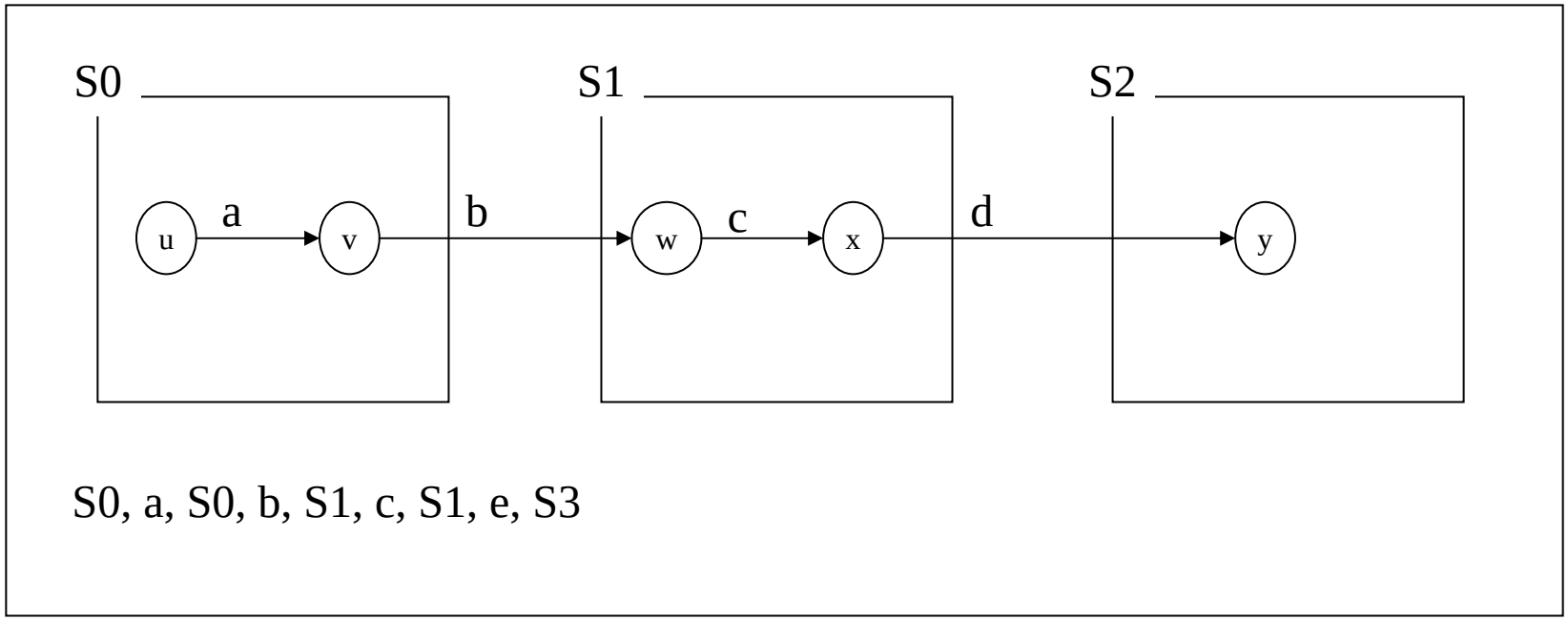
Leads to false Deadlock



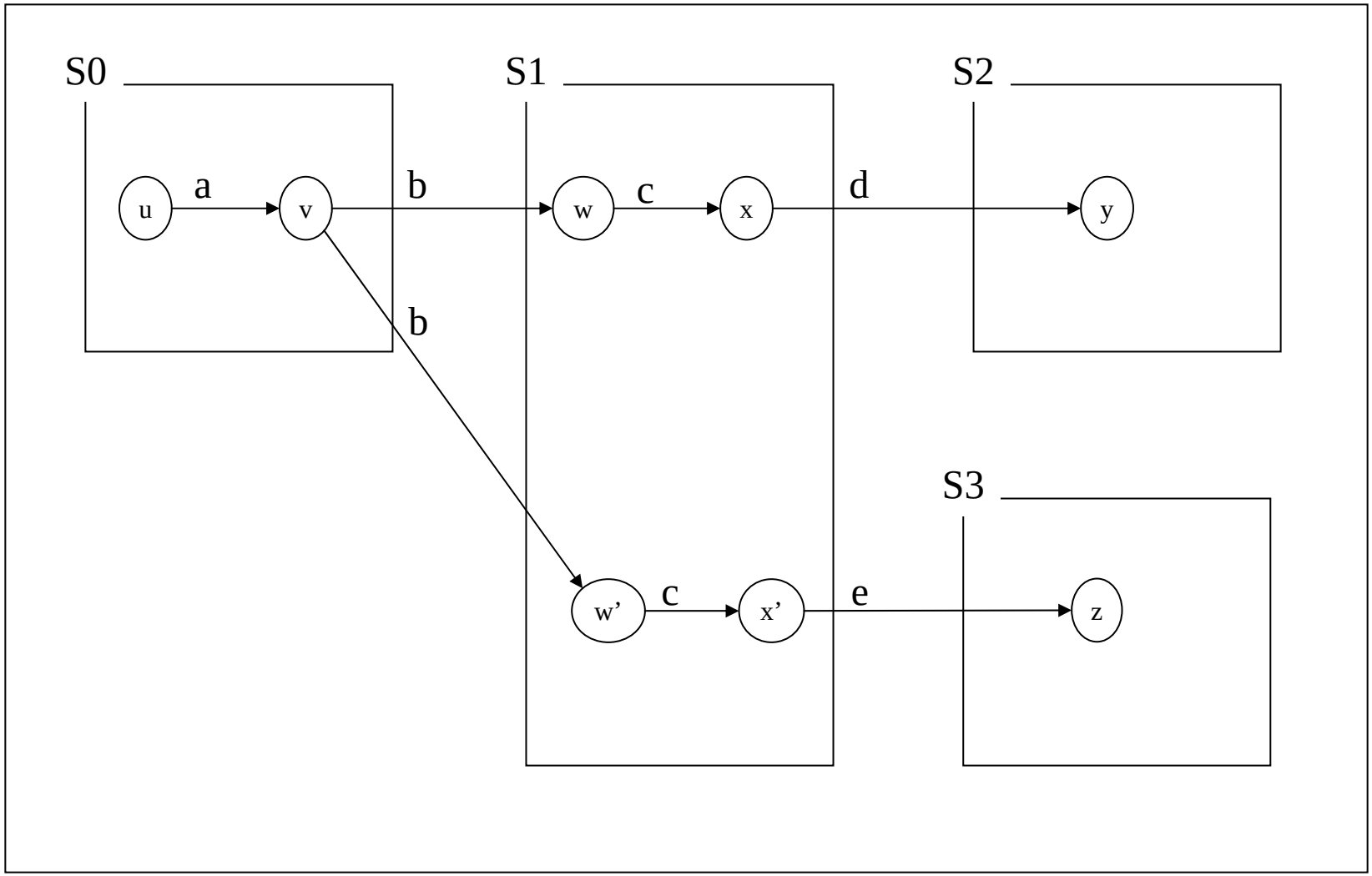
Example Deadlock Avoided



Initial Phase Trace Precursor



Initial Phase Trace Precursor 2



Initial Phase Trace Precursor 3