Specification

- Detailed and precise proposal for a system
- Provides the technical basis for a contract
- Typically increases understanding and causes some revision in the analysis
- Ideally, a specification should:
  - enable clients to validate the system (solve the right problem)
  - establish a basis for developers to verify the system (solve the problem right)

Validation and Verification

- Both hard to achieve in practice
- Validation
  - JAD (Joint Application Development)
  - prototyping
- Verification
  - typical: thoughtful inspection and testing
  - possible: verifiable transformations that preserve
    - information
    - constraints
    - behavior
Formalism

- Advantages
  - requires careful thought
  - provides precision
  - removes unstated assumptions
  - makes correctness proofs possible
  - serves as a basis for tool development
  - enables prototyping
- Disadvantages
  - hard to do and hard to read and understand
  - may hinder productivity

Tunable Formalism

- Various levels of formalism
  - Completely formal must be possible.
  - Completely informal should also be possible.
- Various levels of completion
- System components can vary in their level of completion and formalism.
- OSM supports tunable formalism.
An Approach to Specification

- Establish a system automation boundary.
  - Allow only interface interactions to cross the boundary.
  - Split active boundary-crossing object sets.
  - Note: subsystems may also be specified with an automation boundary.

- Formalize behavior specifications.
  - Tune the formalism of each component appropriately.
  - Scale up specification size and detail with OSM-L.

- Formalize boundary-crossing interactions.
  - Add details about information passed in and out.
  - Use interface forms to lay out and simplify interfaces.

System Automation Boundary

- Restricted high-level object set
  - standard high-level object set
  - only interactions cross the boundary

- Often easy to establish – when:
  - All object and relationship sets are to be in the database.
  - All states and transitions are to be implemented.
  - All interactions are either internal or have either only an origin or destination outside the system.

- Sometimes requires transformations
Interaction Transformations

Chapter 8 - 7

Relationship-Set Transformations

Chapter 8 - 8
Boundary-Crossing Active Object Set

Transformed Active Object Set
Mitosis

- Establish an inside and an outside object set.
- Identify roles for inside and outside object sets.
- Identify synchronization interactions needed to coordinate the activities of the inside and outside object sets.
- Write the state nets for the two object sets and the boundary-crossing interactions between them.

OSM-L
A Formal Specification Language

- Textual Language
  - scales up
  - allows more precision
  - gets us closer to implementation
- Model-Equivalent
  - OSM and OSM-L constructs match one for one
  - analysis work translates directly (seamless)
  - a return to graphical notation is possible
  - mixed OSM/OSM-L is possible and common
OSM-L: Declarations

object "J's BandB";
Room [n];
"J's BandB" [n] has Room [1];
Guest [0:*] has preference for Room [0:*]
| Room is favorite of Guest;
Current Guest, Future Guest isa[union] Guest;

OSM-L: High-Level Declarations

Room includes Room [1] has RoomNr: String [1];
Room [1] has Name: String [a:1]; end;
Guest includes Guest [1] has GuestNr: String [1];
Guest [1] has Name: String [b]; end;
Guest [0:*] has reservation on Arrival Date: String [1:*] for Room [0:*];
Guest(x) has reservation for Room(y) :-
Guest(x) has reservation on Arrival Date() for Room(y);
[ a + b > 0 ];
**OSM-L: Queries**

1. **Predicate calculus** (with text symbols, e.g., $\exists$ is exists).

   GuestNr(x) with Name(y) where exists z exists w (Guest(z) has GuestNr(x) and Guest(z) has Name(y) and Guest(z) has reservation on ArrivalDate(10 May) for Room(w))

2. **Path Expressions.**

   RoomNr(1).Name ArrivalDate(10 May).Guest.Name

---

**OSM-L: State Nets**

Reservation Clerk includes

@ add then enter Ready; end;
when Ready @ remove then end;
when Ready new thread @ new reservation then

...
OSM-L: State Nets

Reservation Clerk includes

- when Ready new thread @ new reservation then
  << request filled-in form >>
  enter Waiting for Form; end;

- when Waiting for Form exception @ cancel then end;

- when Waiting for Form @ form filled then
  << make reservation >>
  exception << form not OK >> enter Error Detected; end;

- when Error Detected then
  << report error; provide partially filled-in form >>
  enter Waiting for Form; end;

- when Ready new thread if << later than 6:00 pm
  and Guest not registered and someone else wants room >> then
  << cancel reservation >> end;

end;

OSM-L: Updates

1. Add and remove.

- add Room
- remove Guest(x) where Guest(x) has GuestNr(111)
- add Guest(x) has reservation on ArrivalDate(10 May) for Room(y)
  where Room(y) has RoomNr(1)

2. Assignment Statements.

- RoomNr(1).Name := Clinton
- RoomNr(5).Name := GuestNr(111).Name
- RoomNr(5) := RoomNr(5)+1
OSM-L: Interactions

tell Guest ("Repair done", Room#) from Proprietor to Reservation Clerk
("The repair you requested is done.") from Reservation Clerk to Guest(x)
where << Guest in Room 1 >>
new reservation to Reservation Clerk
("Please fill in the form", Form) -> (Form) from Reservation Clerk

Note: In context, neither from nor to is needed.

OSM-L: Control Structures

time to check for Special Guests
for each Special Guest(x) do
  if Guest(x) occupies Room() then
    special guest notification (Guest(x).Name, Guest(x).RoomNr);
  end;
end;

OSM-L: Parameters and Local Variables

Reservation Clerk

@ f (x: String, y: Guest, z: Integer)

w: Integer;
A [1:*] is related to B [1:*];
while z < w do ...

Functional Specification

- Elucidate and answer questions (inherent in high-level natural language statements)
- Tunable formalism lets us to choose what to formalize and how much to formalize.
- Efficiency considerations need not concern us (until later, during design).
- Systematic approach to specification
  - identify informal components (triggers, actions, constraints, interactions) needing formalization and formalize them
  - use rapid prototyping (state nets are “executable”)

Sample Unanswered Questions

- What information is on the form?
- What does it mean for the form to be not OK?
- What information, besides the information on the form, do we need to make a reservation?
- How do we get this other information?
- Should we enforce the soft real-time constraint?
- What information do we return to the person?

Sample Formalization

Notes: 1. There are more complex formalizations.
2. Some components are still not fully formal (get available rooms, make reservation, get NextGuestNr).
Interaction Formalization

GuestNr Updater [1] includes
  @ get nextGuestNr() -> (nr: GuestNr) then
    nr := nextGuestNr;
    nextGuestNr := nextGuestNr+1;
  end;
end;

Reservation Maker [1] includes
  @ make reservation (g: GuestNr, n: Name, s: StreetNr, c: City, a: ArrivalDate, d: NrDays, r: RoomNr) then
    newGuest: Guest;
    add Guest(newGuest);
    add Guest(newGuest) has GuestNr(g);
    add Guest(newGuest) with Name(n) lives on StreetNr(s) in City(c);
    add Guest(newGuest) has reservation for Room(x) on ArrivalDate(a) for NrDays(d) where Room(x) has RoomNr(r);
  end;
end;

Form Interface: Insertion

@ make reservation (add)
Guest _______ (new)
GuestNr _______ (new) Name _______
StreetNr _______ City _______
ArrivalDate _______ NrDays _______
Room(x) _______ (connect only)
RoomNr(y) _______ (connect only) [ Room(x) has RoomNr(y) ]
Chapter 8 - 27

Form Interface: Retrieval

@ get available rooms
(input)
ArrivalDate(a) _______ NrDays(d) _______
(output)
AvailableRooms(x)

[ not( exists y exists z exists w exists v ( Room(y) has RoomNr(x) and
Guest(z) has reservation for Room(y) on ArrivalDate(w) for NrDays(v) and
((w <= a and a < w+v) or (a < w and w < a+d)))
]

Chapter 8 - 28

Form Interface: Deletion

@ cancel reservation
(input)
GuestNr _______
(remove)
Guest
GuestNr
(keep)
Room
Form Interface: Modification

@ change address
(input)
GuestNr ________
(modify)
StreetNr ________
City ________