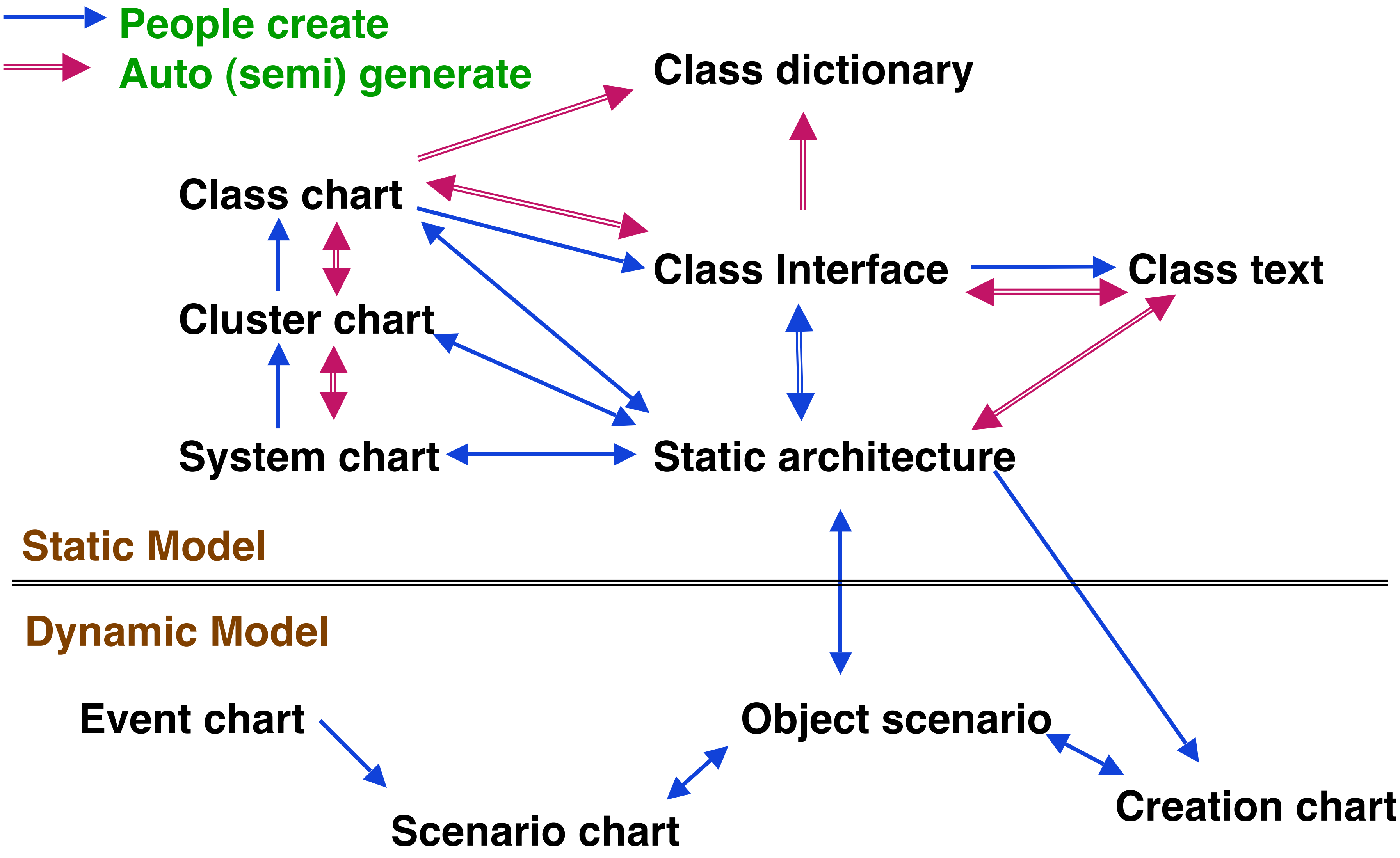
BON
Design Process
(The Method)

Based on slides
by Prof. Paige

BON Process (The Method)

- Process for analysis and development
- Idealized
  - » **In practice it is subject to variation, iteration, reversibility**
- Three phases
  - » **Gathering – collaboration with users**
  - » **Designing – initial working system**
  - » **Refining – improve design, refactor**

BON deliverables & dependencies



People create

Auto (semi) generate

Class dictionary

Class chart

Class Interface

Class text

Cluster chart

System chart

Static architecture

Static Model

Dynamic Model

Event chart

Object scenario

Scenario chart

Creation chart

Gathering Phase Steps 1 & 2

- Delineate the system boundary
  - » **Determine what the system includes and excludes**
  - » **Determine user metaphors**
  - » **Determine the major subsystems**
    - > **Charts: system, event, scenario**
- List candidate classes
  - » **Produce first pass list of classes**
    - > **Charts: class**

Gathering Phase Step 3

- Select classes and groups
  - » **Organize classes into logical groups / clusters**
    - > **Charts: cluster, class**
    - > **class dictionary**
  - » **Determine status of classes**
    - > **Deferred, effective, reused, …**

Example System Chart

System	Conference Management System	Part #
Purpose Conference administration support		Indexing

Cluster	Description
ORGANIZATION	Handles major events occurring during the conference from initial decisions through to conclusion
TECHNICAL_EVENTS	Responsible for putting together the programme, recording status of contributions, checking in reviews and following a precise timetable of what is to be done
REGISTRATION	Collect registration data, produce lists, print badges, send form letters. Store data relevant to whatever may change the cost/benefit of the conference

One per system

Example Cluster Chart

CLUSTER	REGISTRATION	Part #
Purpose: Track conference participants		Indexing

Cluster components	Description
REGISTRATION	Track participant status
ATTENDEE	Track fees and events
REFEREE	Track papers and results
CONTRIBUTOR	Track papers from initial offer to presentation

Similar to the System Chart

Example Class Chart

CLASS	CITIZEN	Part #
Type of Object	Born or resident in a country	Indexing
Queries	Name, Sex, Age, Single, Spouse, Children, Parents	
Commands	Marry, Divorce	
Constraints	• Each citizen has two parents • At most one spouse is allowed • May not marry children or parents • Spouse's spouse must be this person • All children, if any, must have this person as their parent	

Modeling Chart Uses

- Informal charts are useful for exchanging ideas with non-technical people

- Useful for serving as high-level documentation and as a scratch pad for ideas and thoughts

- Idea is to provide medium for social communication and discussing their ideas

Modeling Chart Contents

- System chart
  - » **Exactly one per system**
  - » **Contains a brief description of each top level cluster in the system**
- Cluster chart
  - » **Brief description of a cluster, each class and sub-cluster within it**
- Class chart
  - » **Informally specify class interface.**
    - > **What information and services can other classes ask from the class?**
    - > **What rules must be obeyed by the class?**

Designing Phase

- Define class interfaces
  - » **Use graphical and/or textual descriptions**
- Develop static architecture
- Sketch system behaviour – dynamic properties
  - » **Event charts, scenario charts, object scenarios, creation charts**
  - » **Develop dynamic object model**

Definition of events

- A system is a **black box** with behaviour described by responses to stimuli – **system events**

- An **external event** is triggered by something in the outside world over which the system has no control
  - » **terminal input, interrupts**

- An **internal event** is triggered by the system itself

Dynamic Model Charts

- Event chart
  - » **Lists selected external events that may trigger object communication**
- Scenario chart
  - » **Describes a sequence of events for communicating objects**
- Object creation chart
  - » **Describes which classes create instances of other classes**

Event Chart Example

EVENTS	CONFERENCE_SUPPORT	Part #
Comment	Selected external events triggering representative types of behaviour	Indexing

External	Involved object types
Request to register a submitted paper	CONFERENCE, PROGRAM_COMMITTEE, PAPER

Scenario Chart Example

SCENARIO	DRIVING_SYSTEM	Part #
Comment Borrow car and go for a drive		Indexing

Step 1:

Driver gets keys from owner

DRIVER calls OWNER : send request receive keys

Step 2:

Driver turns ignition

DRIVER calls IGNITION : send turn_on receive NIL

Step 3:

Engine starts

IGNITION calls ENGINE : send turn_on receive NIL

Creation Chart Example

CREATION	MATRIX_SYSTEM	Part #
Comment	Only those classes dealing with the CIRCUS cluster	Indexing

Class	Creates instances of
SPARSE_MATRIX	ARRAY, MATRIX_ELEMENT
MATRIX_ELEMENT	STACK [ELEPHANT]
MINIMUM_TEST	SPARSE_MATRIX, MATRIX_ELEMENT, STRING, ELEPHANT

Refining Phase

- Refine system
  - » **Find new design classes, add new features**
    - > **Modify: Class interfaces, static architecture, class dictionary, event charts, object scenarios**
- Generalize
  - » **Factor out common behaviour**
    - > **Modify: class interfaces, static architecture, class dictionary**
- Complete and review system
  - » **Produce final static architecture with dynamic system behaviour**
    - > **All deliverables complete**

Software Development Methods

- Many good ideas and much effort put into producing recipes for constructing software
  - » **But no sure fire method**
- No easy path to producing quality software
  - » **F.P. Brooks Jr., *No Silver Bullet* , Computer, Vol. 20, No. 4, April 1987, pp. 10..19.**
  - » **Replies in Computer, Vol. 20, No. 7, July 1987, pp 7..9.**
- As our knowledge and experience have increased so has our reach

Understand Limitations and Benefits

- General principles for constructing software can be taught
  - » **But no teaching can guarantee success**
- This is not to say methods are worthless
  - » **If you restrict their domain of applicability, you can have success**
- Many method creators are unwilling to do this
  - » **They want to sell their method – and its tools**
- All relies on invention, creativity and expertise of the individual developers