Nguyên mẫu phần mềm

Phát triển ứng dụng nhanh
để xác minh yêu cầu

©Ian Sommerville 2000

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Mục tiêu

- Mô tả việc sử dụng prototype trong các loại dự án phát triển khác nhau
- Thảo luận việc tạo ra nguyên mẫu tiến hóa và dùng một lần rồi bỏ
- Giới thiệu 3 kĩ thuật prototype – phát triển ngôn ngữ mức cao, lập trình CSDL và tái sử dụng các thành phần
- Giải thích nhu cầu tạo nguyên mẫu giao diện

Các chủ đề đề cập

- Prototyping trong tiến trình phần mềm
- Các kĩ thuật prototyping
- Prototyping giao diện người dùng

Prototyping hệ thống

- Prototyping là cách phát triển nhanh của một hệ thống
- Trong quá khứ, hệ thống được phát triển được nghĩ là kém hơn theo cách nào đó với hệ thống được yêu cầu vì vậy cần phải phát triển thêm
- Giờ đây, biên giữa prototype và hệ thống bình thường được xóa nhòa và trở thành một cách tiếp cận tiến hóa

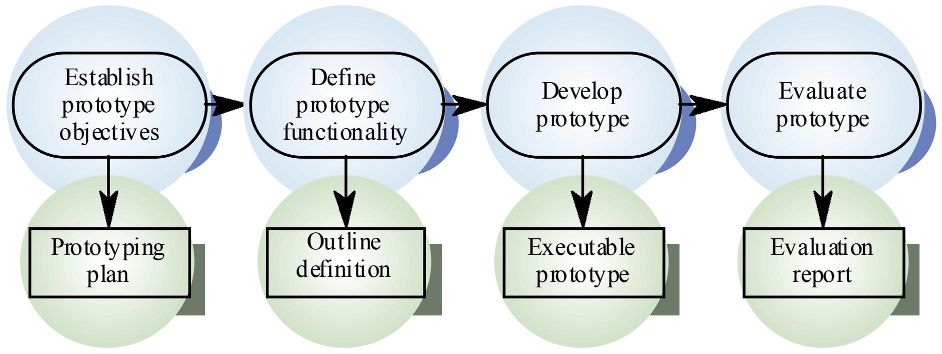
Sử dụng prototype hệ thống

- Nguyên tắc sử dụng là khách hàng và nhà phát triển hiểu yêu cầu của hệ thống
  - Phát hiện yêu cầu. Người dùng có thể trải nghiệm với prototype để xem thử hệ thống hỗ trợ công việc của họ ra sao
  - Kiểm tra tính hiệu lực của yêu cầu. Prototype can reveal errors and omissions in the requirements
- Prototyping can be considered as a risk reduction activity which reduces requirements risks

Prototyping benefits

- Misunderstandings between software users and developers are exposed
- Missing services may be detected and confusing services may be identified
- A working system is available early in the process
- The prototype may serve as a basis for deriving a system specification
- The system can support user training and system testing

Prototyping process



Prototyping benefits

- Improved system usability
- Closer match to the system needed
- Improved design quality
- Improved maintainability
- Reduced overall development effort

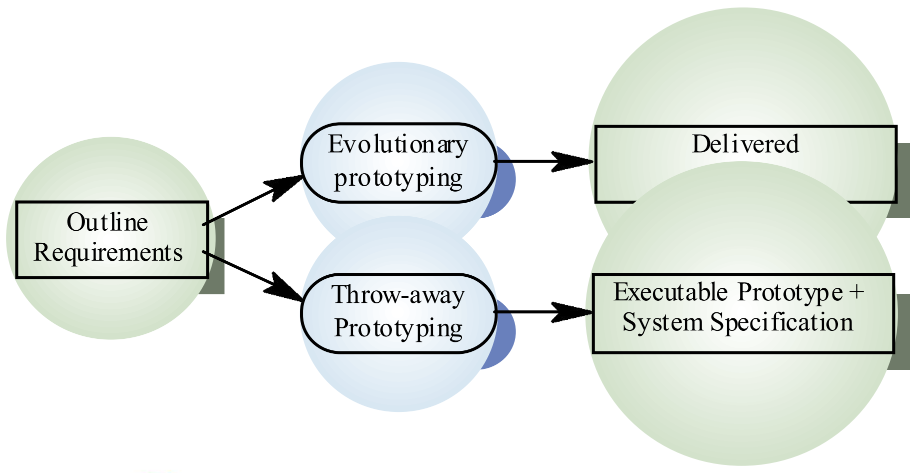
Prototyping in the software process

- Evolutionary prototyping
  - An approach to system development where an initial prototype is produced and refined through a number of stages to the final system
- Throw-away prototyping
  - A prototype which is usually a practical implementation of the system is produced to help discover requirements problems and then discarded. The system is then developed using some other development process

Prototyping objectives

- The objective of *evolutionary prototyping* is to deliver a working system to end-users. The development starts with those requirements which are best understood.
- The objective of *throw-away prototyping* is to validate or derive the system requirements. The prototyping process starts with those requirements which are poorly understood

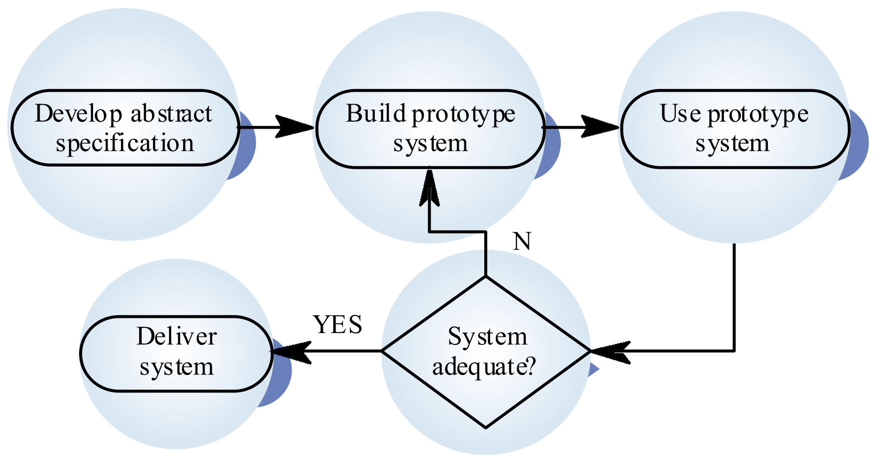
Approaches to prototyping



Evolutionary prototyping

- Must be used for systems where the specification cannot be developed in advance e.g. AI systems and user interface systems
- Based on techniques which allow rapid system iterations
- Verification is impossible as there is no specification. Validation means demonstrating the adequacy of the system

Evolutionary prototyping



Evolutionary prototyping advantages

- Accelerated delivery of the system
  - Rapid delivery and deployment are sometimes more important than functionality or long-term software maintainability
- User engagement with the system
  - Not only is the system more likely to meet user requirements, they are more likely to commit to the use of the system

Evolutionary prototyping

- Specification, design and implementation are inter-twined
- The system is developed as a series of increments that are delivered to the customer
- Techniques for rapid system development are used such as CASE tools and 4GLs
- User interfaces are usually developed using a GUI development toolkit

Evolutionary prototyping problems

- Management problems
  - Existing management processes assume a waterfall model of development
  - Specialist skills are required which may not be available in all development teams
- Maintenance problems
  - Continual change tends to corrupt system structure so long-term maintenance is expensive
- Contractual problems

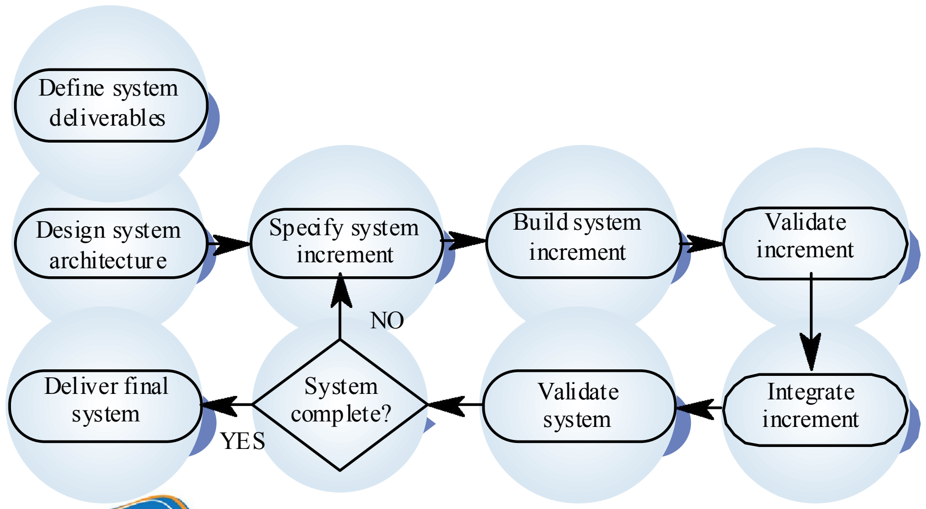
Prototypes as specifications

- Some parts of the requirements (e.g. safety-critical functions) may be impossible to prototype and so don’t appear in the specification
- An implementation has no legal standing as a contract
- Non-functional requirements cannot be adequately tested in a system prototype

Incremental development

- System is developed and delivered in increments after establishing an overall architecture
- Requirements and specifications for each increment may be developed
- Users may experiment with delivered increments while others are being developed. therefore, these serve as a form of prototype system
- Intended to combine some of the advantages of prototyping but with a more

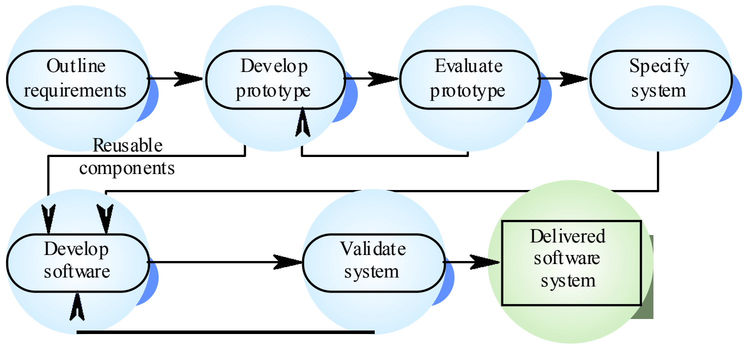
Incremental development process



Throw-away prototyping

- Used to reduce requirements risk
- The prototype is developed from an initial specification, delivered for experiment then discarded
- The throw-away prototype should NOT be considered as a final system
  - Some system characteristics may have been left out
  - There is no specification for long-term maintenance
  - The system will be poorly structured and difficult to maintain

Throw-away prototyping



Prototype delivery

- Developers may be pressurised to deliver a throw-away prototype as a final system
- This is not recommended
  - It may be impossible to tune the prototype to meet non-functional requirements
  - The prototype is inevitably undocumented
  - The system structure will be degraded through changes made during development
  - Normal organisational quality standards may not have been applied

Rapid prototyping techniques

- Various techniques may be used for rapid development
  - Dynamic high-level language development
  - Database programming
  - Component and application assembly
- These are not exclusive techniques - they are often used together
- Visual programming is an inherent part of most prototype development systems

Dynamic high-level languages

- Languages which include powerful data management facilities
- Need a large run-time support system. Not normally used for large system development
- Some languages offer excellent UI development facilities
- Some languages have an integrated support environment whose facilities may be used in the prototype

Prototyping languages

Language	Type	Application domain
Smalltalk	Object-oriented	Interactive systems
Java	Object-oriented	Interactive systems
Prolog	Logic	Symbolic processing
Lisp	List-based	Symbolic processing

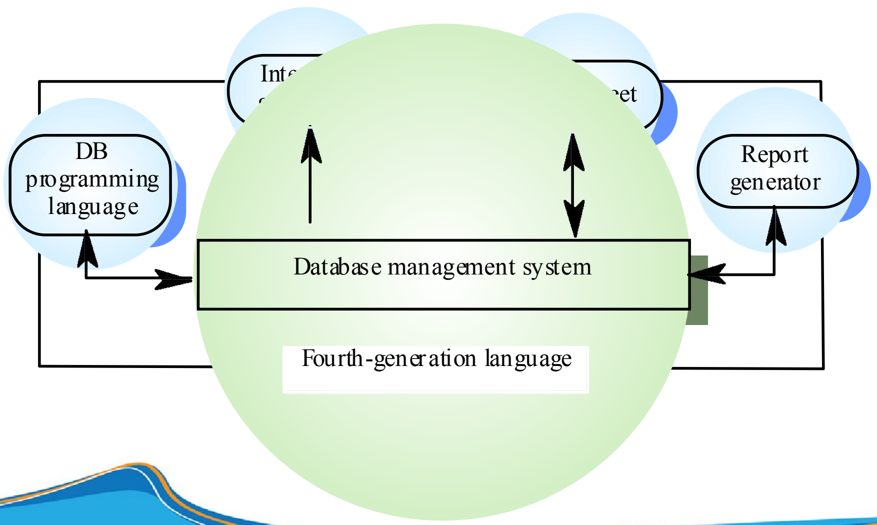
Choice of prototyping language

- What is the application domain of the problem?
- What user interaction is required?
- What support environment comes with the language?
- Different parts of the system may be programmed in different languages. However, there may be problems with language communications

Database programming languages

- Domain specific languages for business systems based around a database management system
- Normally include a database query language, a screen generator, a report generator and a spreadsheet.
- May be integrated with a CASE toolset
- The language + environment is sometimes known as a fourth-generation language (4GL)
- Cost-effective for small to medium sized

Database programming



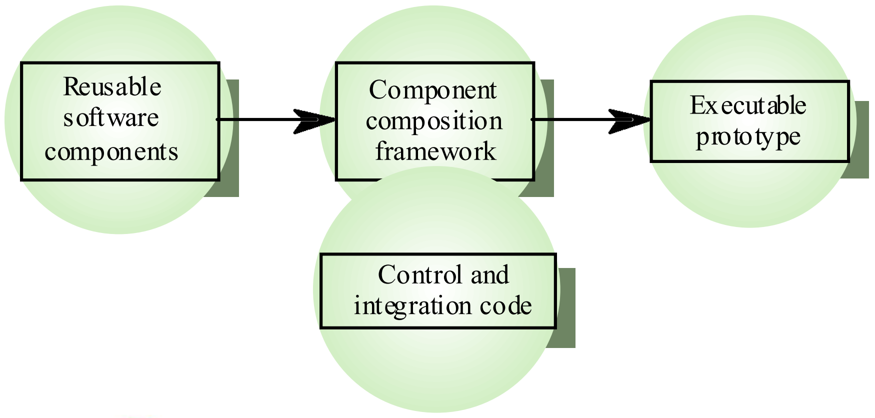
Component and application assembly

- Prototypes can be created quickly from a set of reusable components plus some mechanism to ‘glue’ these component together
- The composition mechanism must include control facilities and a mechanism for component communication
- The system specification must take into account the availability and functionality of existing components

Prototyping with reuse

- Application level development
  - Entire application systems are integrated with the prototype so that their functionality can be shared
  - For example, if text preparation is required, a standard word processor can be used
- Component level development
  - Individual components are integrated within a standard framework to implement the system
  - Frame work can be a scripting language or an integration framework such as CORBA

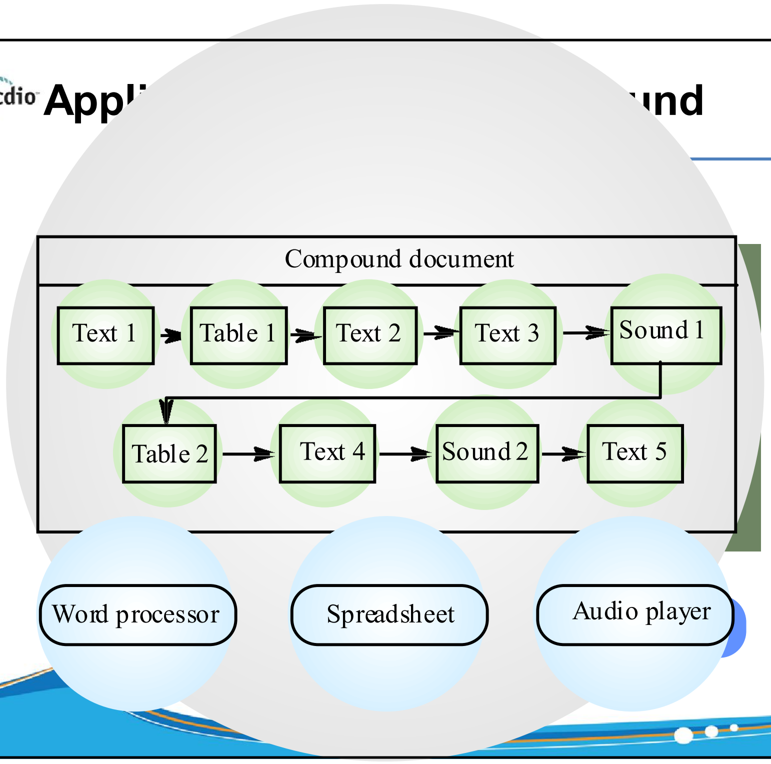
Reusable component composition


Reusable software components
Component composition framework
Executable prototype
Control and integration code

Compound documents

- For some applications, a prototype can be created by developing a compound document
- This is a document with active elements (such as a spreadsheet) that allow user computations
- Each active element has an associated application which is invoked when that element is selected
- The document itself is the integrator for the different applications

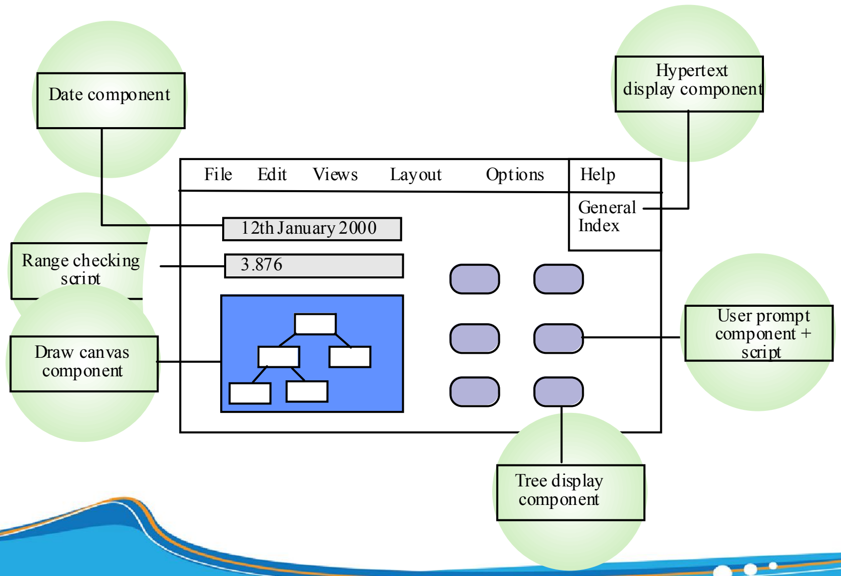
Appl [illegible] und



Visual programming

- Scripting languages such as Visual Basic support visual programming where the prototype is developed by creating a user interface from standard items and associating components with these items
- A large library of components exists to support this type of development
- These may be tailored to suit the specific application requirements

Visual programming with reuse



Problems with visual development

- Difficult to coordinate team-based development
- No explicit system architecture
- Complex dependencies between parts of the program can cause maintainability problems

User interface prototyping

- It is impossible to pre-specify the look and feel of a user interface in an effective way. prototyping is essential
- UI development consumes an increasing part of overall system development costs
- User interface generators may be used to ‘draw’ the interface and simulate its functionality with components associated with interface entities
- Web interfaces may be prototyped using a web site editor

Types of prototype

- Business prototypes
- Usability prototypes
- Performance and capacity
- Capability/technique prototypes

Key points

- A prototype can be used to give end-users a concrete impression of the system's capabilities
- Prototyping is becoming increasingly used for system development where rapid development is essential
- Throw-away prototyping is used to understand the system requirements
- In evolutionary prototyping, the system is developed by evolving an initial version to the final version

Key points

- Rapid development of prototypes is essential. This may require leaving out functionality or relaxing non-functional constraints
- Prototyping techniques include the use of very high-level languages, database programming and prototype construction from reusable components
- Prototyping is essential for parts of the system such as the user interface which cannot be effectively pre-specified. Users must be involved in prototype evaluation