CS5351 Software Engineering
2024/25 Semester A
Software Maintenance

Dr W.K. Chan

Department of Computer Science

Email: **wkchan@cityu.edu.hk**

Website: **http://www.cs.cityu.edu.hk/~wkchan**

Purposes of S/W Maintenance

The *purposes* of performing maintenance are to

- provide continuity of service, keep it operational
  - fix bugs, recover from failures, respond to environments
- support mandatory upgrade
  - respond to regulation changes
  - maintain competitive edge
- support user requests for improvement
  - customize, enhance functionality, improve performance
- facilitate future maintenance work
  - e.g., perform code restructuring, documentation update

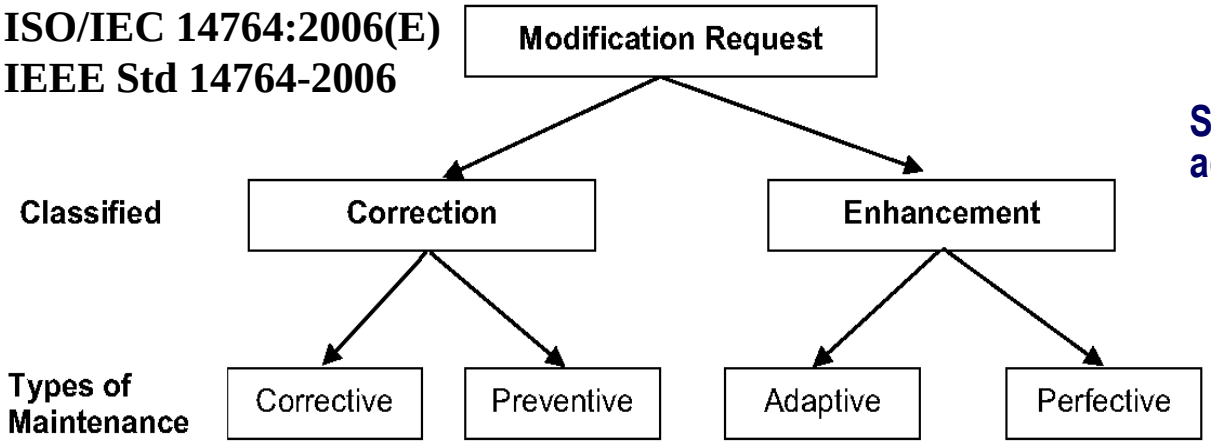
Characteristics of the Maintenance Phase

- **[People] Staff expertise**
  - Most essential during initial development and evolution (can be AI-assisted)
- **[Product] Software architecture**
  - Most stable at the time of the first release version
  - Gradually degrades due to restructuring for changes
- **[Process] Software decay**
  - Quality of design and code deteriorates, docs outdated
  - Ad hoc fixes avoids major changes but less maintainable
  - Requires increasingly more expertise to maintain
- **[Cost] Economics**
  - Heavy initial investment, gaining increasing benefit up to a max; then trail off, forcing phaseout and closedown
  - To regain benefits, cycle repeats with next development

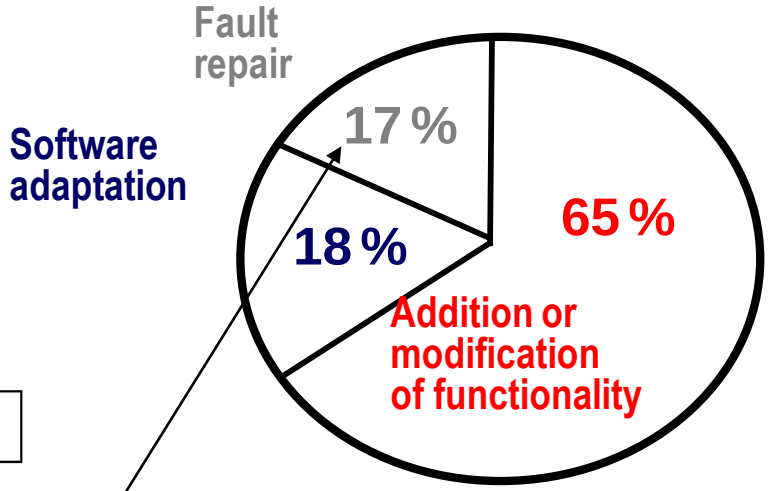
Classification of Modification Requests

- **Correction**: modification to correct a discovered/potential problem
- **Enhancement**: modification to satisfy a new requirement

ISO/IEC 14764:2006(E)
IEEE Std 14764-2006



Distribution of efforts
Sommerville, 2007



AI chatbot
in Nov 2024: 20%-25% 5%-10% 15%-20% 50%-55%

25%-35% increased (vs. 17%)

Trend: efforts shift from Enhancement to Correction

Data in Canada

- Software maintenance costs range between **15% to 25%** annually of the overall software development cost.

Maintenance Type	Approximate Percentage of Total Maintenance Cost	Cost Range (Assuming Original Development Cost of $1M)	Examples
Corrective Maintenance (Bug fixes and defects)	20%	$30,000 – $50,000	Fixing errors in software like Microsoft Office, Adobe Photoshop
Adaptive Maintenance (Adjusting to changes in the environment)	25%	$37,500 – $62,500	Updating an e-commerce app to work with a new version of iOS or Android
Perfective Maintenance (Improvements and enhancements)	50%	$75,000 – $125,000	Adding a new feature to a social media platform like Facebook or Twitter
Preventive Maintenance (Optimization and future-proofing)	5%	$7,500 – $12,500	Refactoring code in a banking system to improve performance and prevent future issues
Total Maintenance Costs	–	**$1,50,000 – $2,50,000**	–

https://www.spaceo.ca/blog/software-maintenance-cost/

4 Type of Software Maintenance

- **Adaptive Maintenance**
  - Modify the software to ensure it continues to operate in a new or changed environment, such as when there are updates to the operating system (e.g., from Android to HarmonyOS Next), hardware, or other software dependencies (e.g., handling depreciated APIs or using a new framework version).
- **Perfective Maintenance**
  - Enhance the software by adding new features or requirements. It aims to improve the software's functionality and performance based on user feedback or new business needs.
  - E.g., for meeting the requirements in a sprint backlog

4 Type of Software Maintenance

- **Corrective Maintenance**
  - Fix bugs and errors that are discovered *after the software has been deployed.*
  - Sometimes, to handle unexpected failures or critical issues that need immediate attention, which involves prompt fixing the software to restore its functionality and minimizing downtime.
- **Preventive Maintenance**
  - Make changes to the software to prevent potential future problems. It aims to improve the software's maintainability and reliability by addressing issues before they become significant problems. (e.g., addressing technical debts)

Their Processes are Different Too

- Adaptive Maintenance
  - **Identify changes in the system environment, analyze the impact on the software**, plan and design modifications, implement changes, test the updated software, and deploy the changes.
- Perfective Maintenance
  - **Gather user feedback to identify areas of improvement**, elicit new requirements, analyze and prioritize enhancements, design and implement new features or improvements, test the enhancements, and deploy them.
- Corrective Maintenance: **Typical** bug fixing process
- Preventive Maintenance
  - Conduct regular code reviews and **system audits**, **identify potential future issues**, implement changes to improve maintainability, **test the preventive measures**

Where to Release Changes

	Adaptive Changes	Perfective Changes	Corrective Changes
Major Release	Yes	Yes	Yes
Minor Release	Small	Yes	Yes
Emergency Release	No	No	Yes

Source: ESA PSS-05-07 Issue 1 Revision 1
THE OPERATIONS AND MAINTENANCE PHASE

How about maintenance in general (not restricted to S/W Maintenance)?

e.g.,

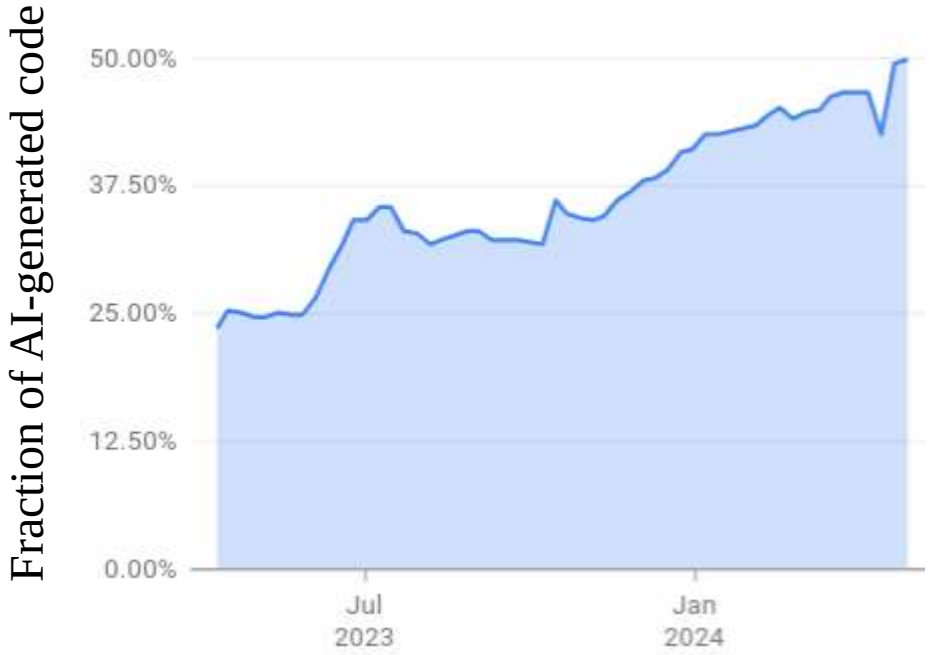
Comparison of maintenance types

Maintenance type	Preventive Maintenance					Corrective Maintenance	
	Time based	Failure finding	Risk based	Condition based	Predictive	Deferred	Emergency
Task type	Scheduled Overhaul / Replacement	Functional Test	Measurement of condition	Calculation and extrapolation of parameters	Inspection or Test	Repair/ Replace	Repair/ Replace
Objective	Restore or replace regardless of condition	Determine if hidden failure has occurred	Restore or replace based on a measured condition compared to a defined standard	Determine if failure is imminent and intervention is required	Determine condition and conduct risk assessment to determine when next inspection, test or intervention is required.	Restore or replace following failure. Result of a Run to Failure Strategy or an unplanned failure.	Restore or replace following unplanned failure.
Interval	Fixed time or usage interval e.g. 1 month, 1,000hrs or 10,000 km	Fixed time interval (can be set based on risk assessment e.g. SIL)	Fixed time interval for condition measurements/ inspections	Continuous online monitoring of parameters, intervention as required	Time based interval between tasks and scope of task is based on risk assessment	Not applicable, but intervention is deferred to allow for proper planning & scheduling.	Immediate intervention required.

Lower theS/W Maintenance Cost. How?

- Apply CI/CD (DevOps)
- Maintain the software maintenance regularly
  - E.g., use fixed durations for code refactoring, minimize TD, up-to-date documentation
  - Ensure to have preventive maintenance (e.g., system audit, capacity planning)
- Thieve for good software development
  - Good practice, good design, strong staff
- Use automation tools: e.g., auto-notification, auto-test
- Use a bug tracking system
- Use modern (reliable) frameworks
- Optimize software license
- Security update, version control, apply change management

What Else is Big Tech doing?

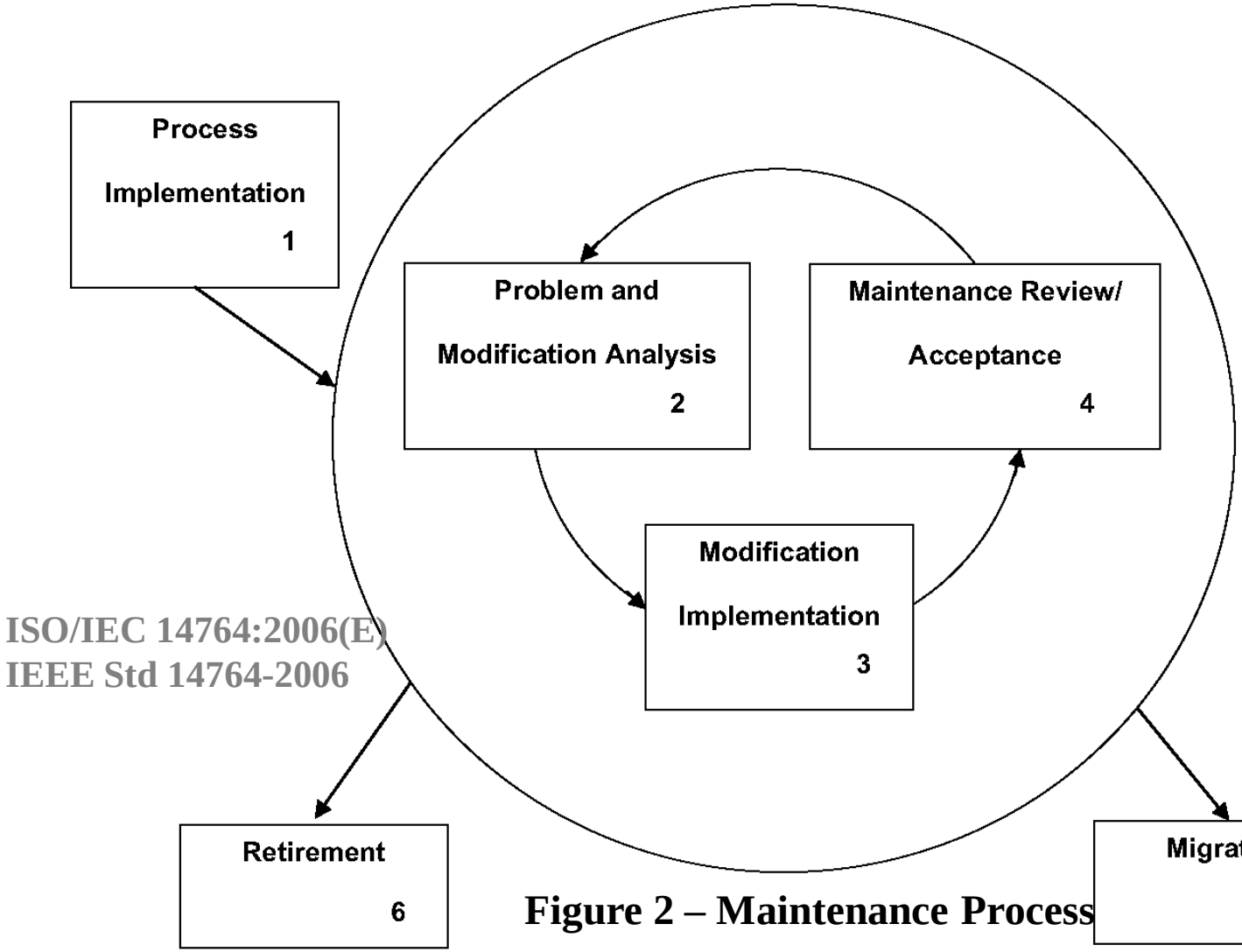


Google [7]

Copyrights issue if using AIgen Code

- Suggested Reading. Read Ref [8]
- Excerpt from Ref [8]
- "If the generated code **includes** open-source **code** publicly **available under a copyleft license**, such as GPL v3, then it may **cause** the entire generated code to **inherit the same open-source license**, which may "taint" **the developer's whole source code**, including proprietary code."
- "when an open-source licensed code is **repeated as generated code** by a generative AI tool, there may be **copyright violation** when the usage of such code does not follow the open-source license terms"
- "whether the developed code may be considered **fair use**, and thereby **avoid copyright violation**, because the copyrighted portion of the source code is **transformed into something with new utility or meaning**."
- "The US Copyright Office has recently decided that **AI cannot be an author of a creative work**." (Note that the US uses the Common Law system)

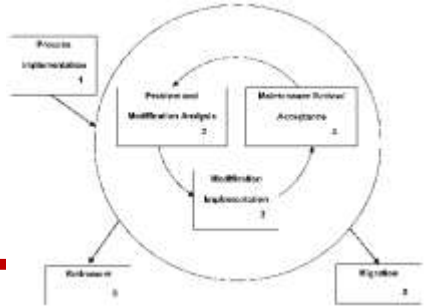
The ISO/IEEE Maintenance Process



ISO/IEC 14764:2006(E)
IEEE Std 14764-2006

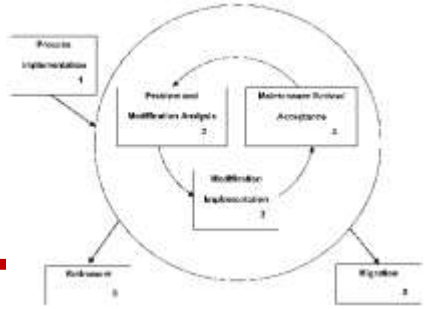
Figure 2 – Maintenance Process

The ISO/IEEE Maintenance Activities



1. **Process Implementation**
   - 1.1 Develop maintenance plans and procedures
   - 1.2 Establish MR/PR procedures
   - 1.3 Invoke the Configuration Management Process
2. **Problem and Modification Analysis**
   - 2.1 Analyze the MR/PR to determine its impact and priority
   - 2.2 Replicate or verify the problem
   - 2.3 Develop options for implementation of modification
   - 2.4 Document MR/PR, analysis results, implementation options
   - 2.5 Obtain approval for the selected modification option
3. **Modification Implementation**
   - 3.1 Conduct analysis, identify the units and docs to be modified
   - 3.2 Invoke the Development Process (a mini-SDLC)

The ISO/IEEE Maintenance Activities



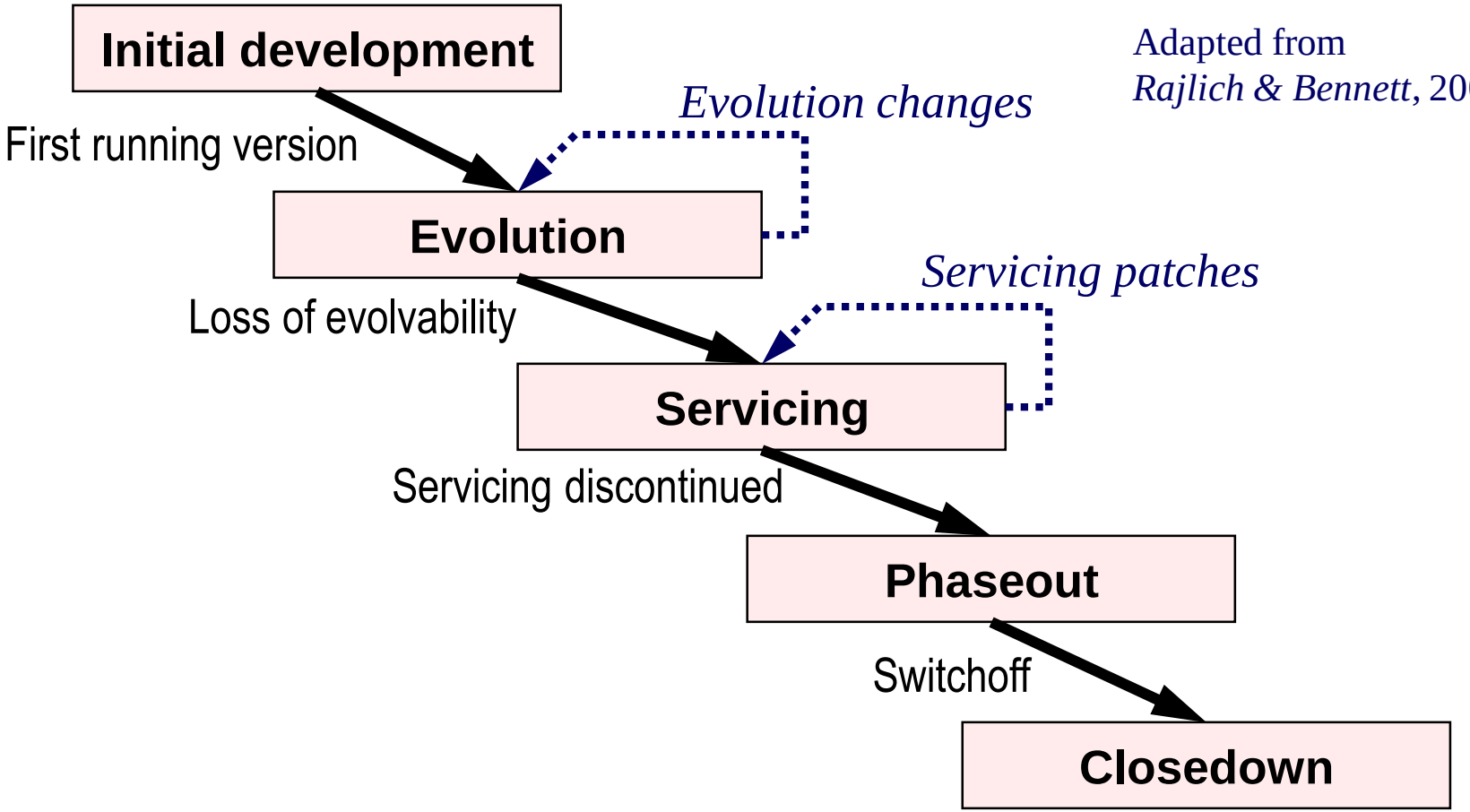
4. **Maintenance Review/Acceptance**
   - 4.1 Conduct review to ensure integrity of modified system
   - 4.2 Obtain approval for satisfactory completion of changes
5. **Migration**
   - 5.1-5.2 Identify modified software/data; Develop migration plan
   - 5.3-5.5 Notify users; Provide training; Notify for completion
   - 5.6 Post-hoc review: to assess impact of new environment
   - 5.7 Archive data (and old software)
6. **Retirement**
   - 6.1 Develop retirement plan
   - 6.2-6.4 Notify users; Implement parallel operations and training; Notify for completion
   - 6.5 Archive data (and old software)

Stages of Maintenance

Within the maintenance phase, there are several stages:

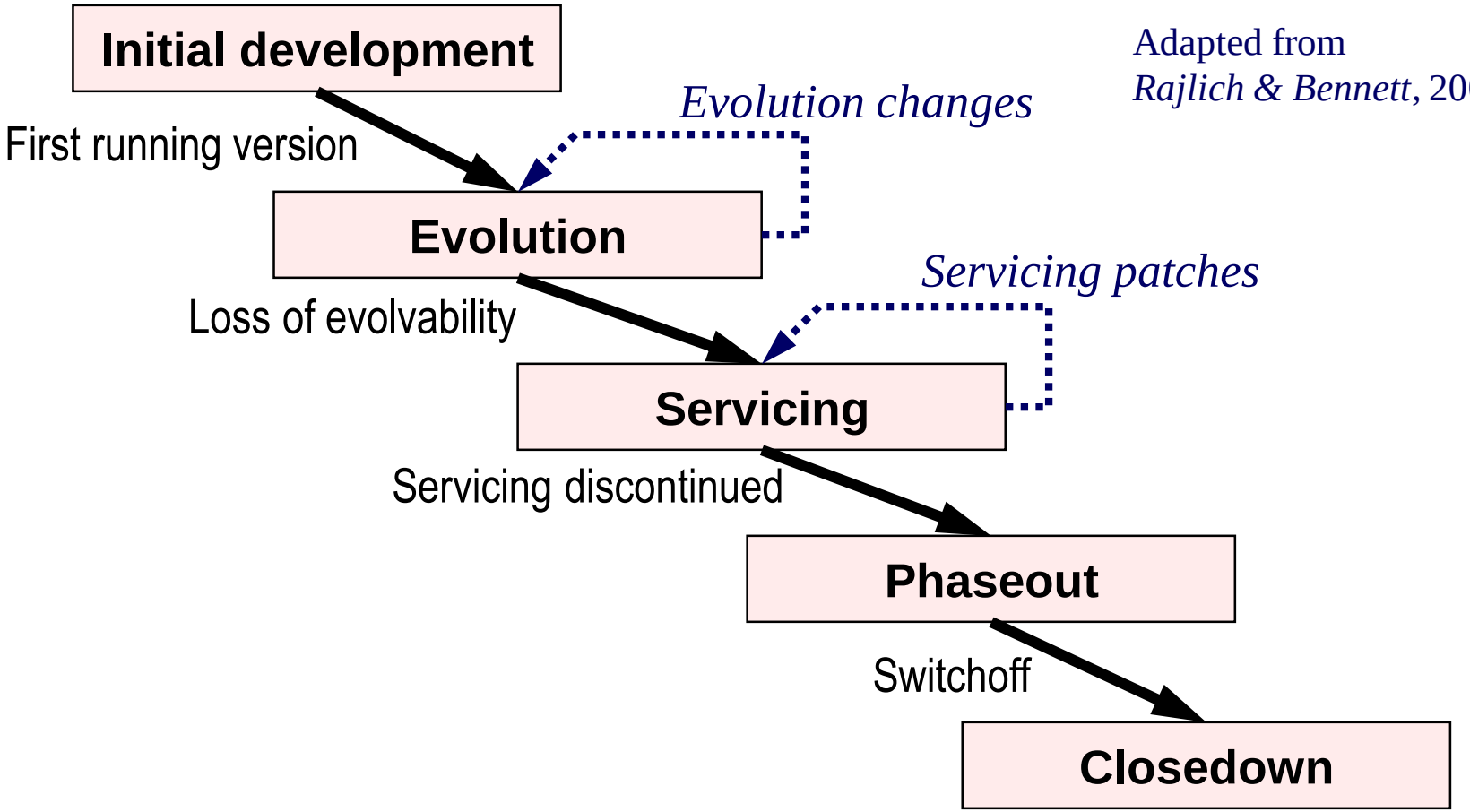
- **Initial development**
  - first release plus accompanying activities (training, etc.)
- **Evolution**
  - major extension of the system's capabilities
- **Servicing**
  - in operation with only minor repairs or simple changes
  - work is stable, mature, well understood
- **Phaseout**
  - services discontinued, though system still in operation
- **Closedown**
  - system withdrawn, users directed to replacement if any

A Simple Staged Model



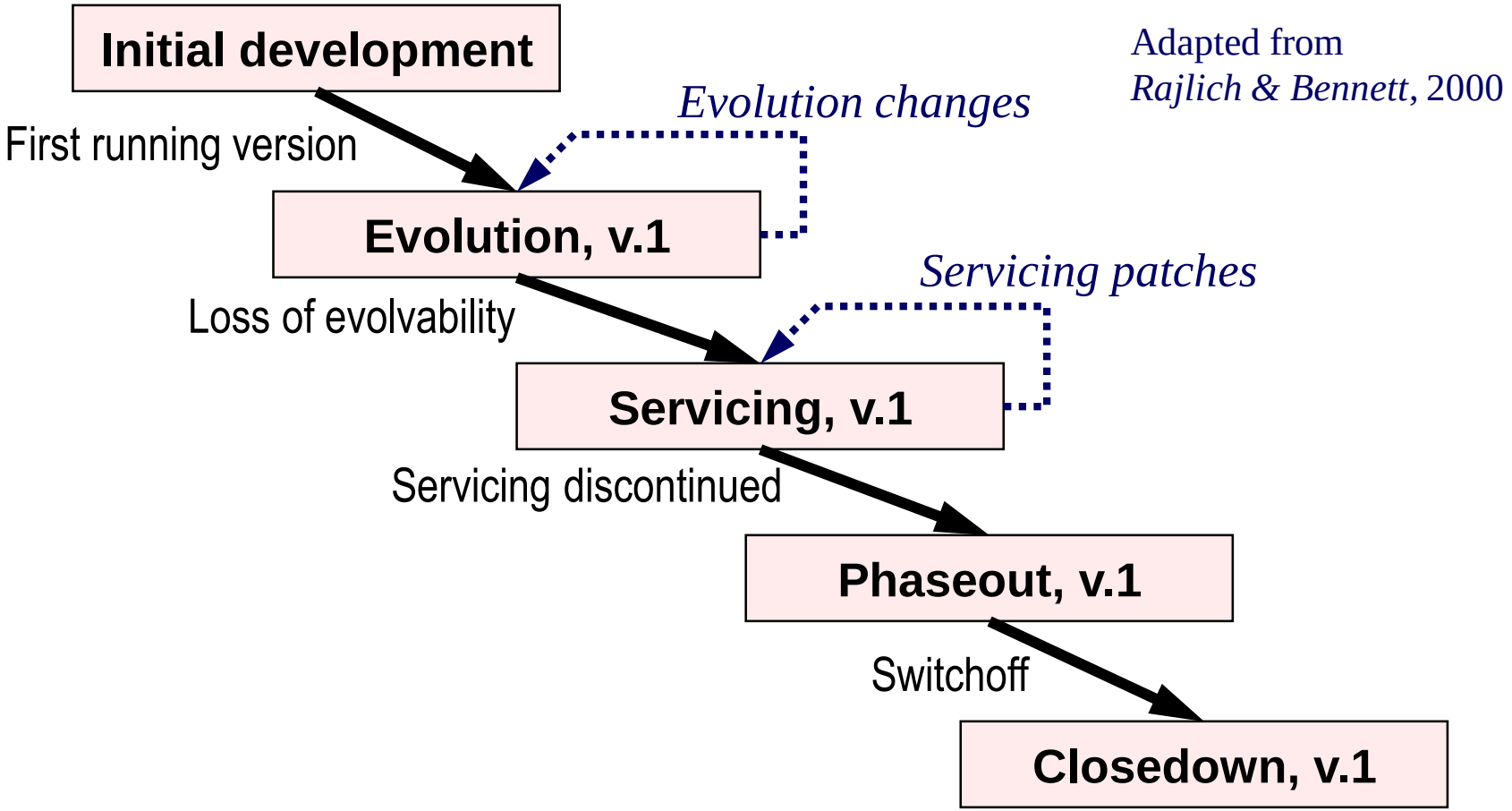
Adapted from
Rajlich & Bennett, 2000

A Simple Staged Model



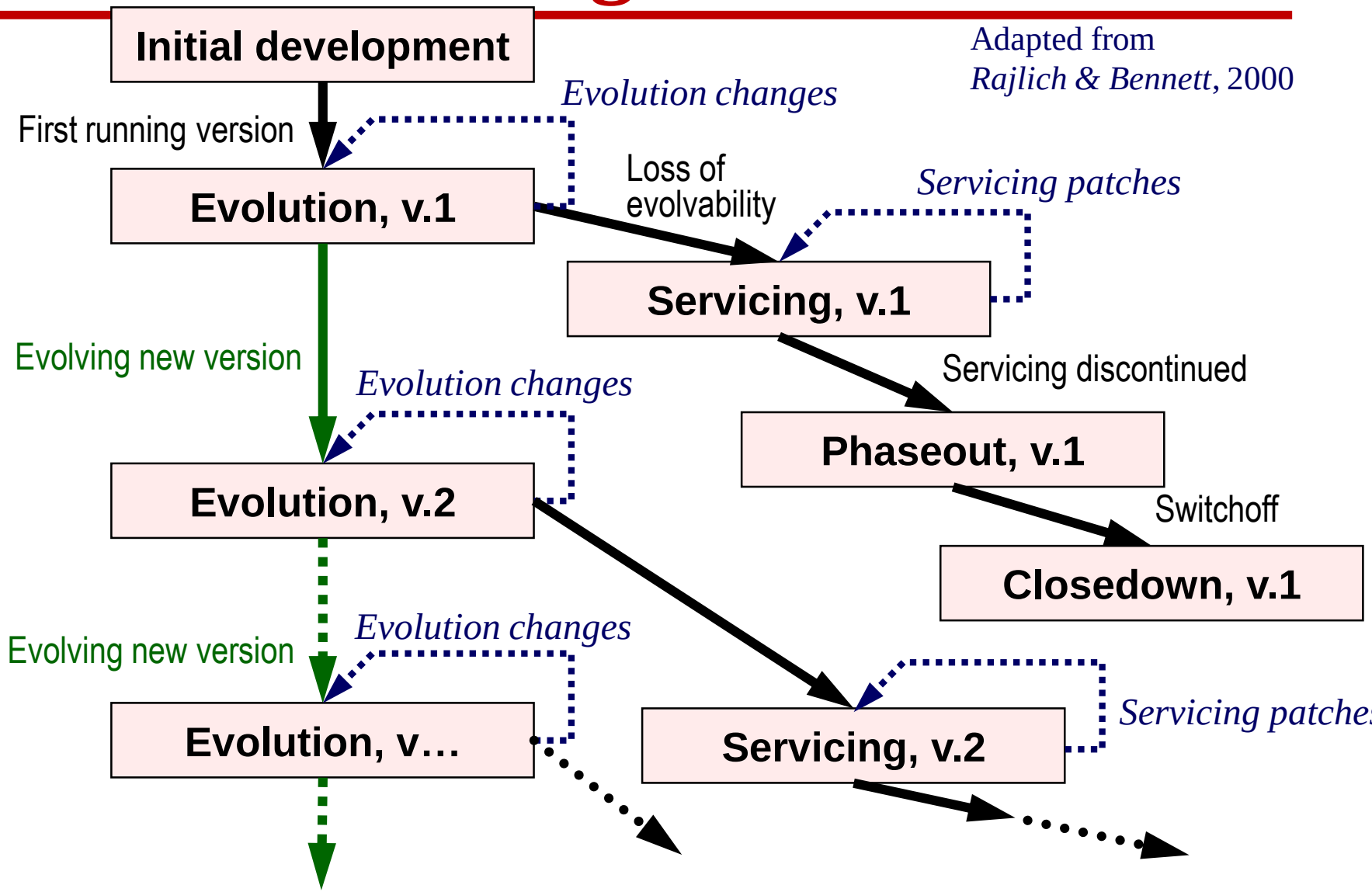
Adapted from
Rajlich & Bennett, 2000

A Versioned Staged Model



Adapted from
Rajlich & Bennett, 2000

A Versioned Staged Model



References

1. V.T. Rajlich & K.H. Bennett (2000). "A Staged Model for the Software Life Cycle". *IEEE Computer*, vol.33, no.7, pp.66-71.
2. ISO/IEEE Std 14764-2006: *Software Engineering – Software Life Cycle Processes – Maintenance*. Sections 3-5.
3. P. Grubb & A.A. Takang (2003). Software Maintenance: Concepts and Practice. 2nd ed., World Scientific. Chapter 1 and Section 3.3.1.
4. ISO/IEC 9126-1:2001(E): *Software engineering – Product quality – Part 1: Quality model*. Section 6.5.
5. I. Sommerville (2007). *Software Engineering*. 8th ed., Addison-Wesley. Section 21.2.
6. S. Schach et al. (2003). Determining the Distribution of Maintenance Categories: Survey versus Measurement. *Empirical Software Engineering* 8(1):351-365.
7. Satish Chandra (2024). AI in software engineering at Google: Progress and the path ahead. Google blog https://research.google/blog/ai-in-software-engineering-at-google-progress-and-the-path-ahead/
8. Bloomberg Law (2023). Copyrights, Professional Perspective - IP Issues With AI Code Generators https://www.bloomberglaw.com/external/document/X4H9CFB4000000/copyrights-professional-perspective-ip-issues-with-ai-code-gener
9. Arkadiusz Krysik (2024) SDLC Guide: A Comprehensive Guide to Effective Software Maintenance Phase https://stratoflow.com/software-maintenance-process/