Software Lifespan

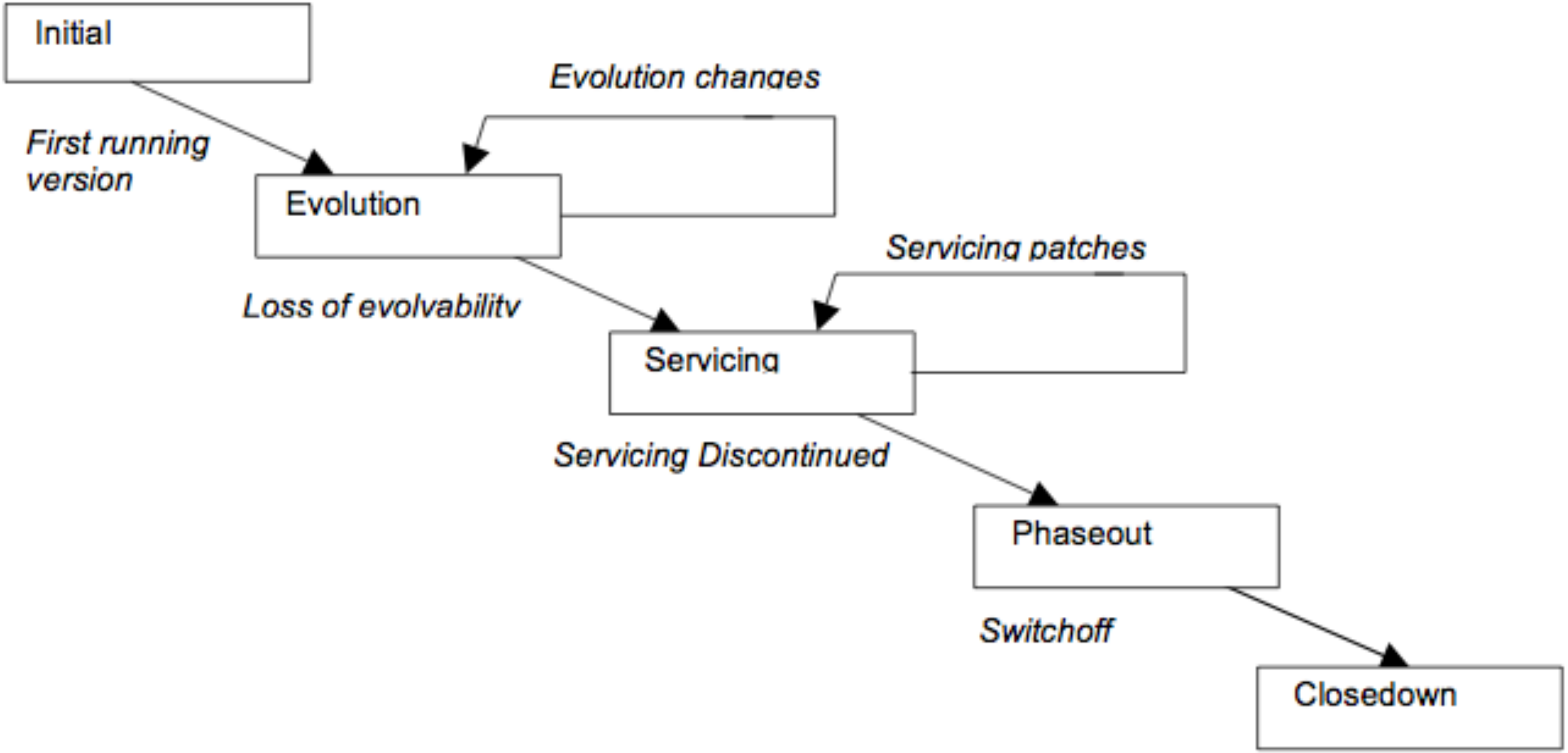
Prof. Romain Robbes

Most software processes are only looking at a part of the picture

They describe initial development, but stop at product delivery, sidestepping many issues in maintenance (or they have a tiny “maintenance” box).

The entire lifespan of a software system may be much longer!

The staged model of software lifespan



Stages of the lifespan

Initial development: requirements, analysis, design, implementation, verification ... you know the drill!

Evolution: new functionality is requested and added to the system ... you know the drill!

Final stages: the software stops evolving and is gradually shut down.

Software enters final stages for several reasons:

Stability: no more large changes are needed.

Code decay: it is too expensive to make changes, a rewrite is needed.

Business reasons: loss of usefulness, better products, etc.

Software in final stages is called “legacy software”.

Final stages

Servicing: only minor changes are made (bug fixes, security issues).
Phaseout: no changes are accepted anymore (help support may still be available).
Closedown: the systems ends its operations, remaing customers must upgrade. Long-lived data must be exported to a newer system.

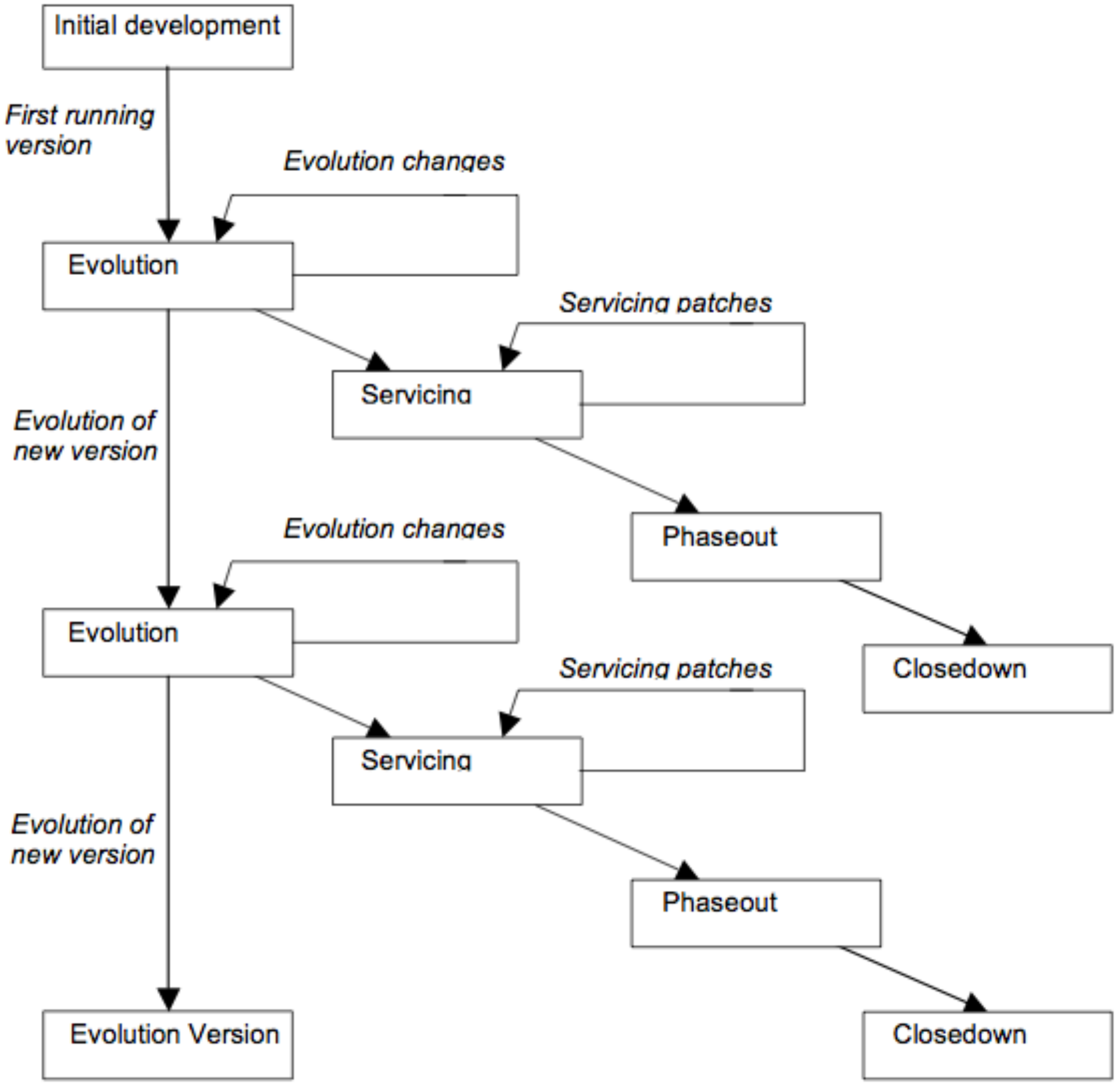
Several versions of the same system can coexist

There can be users of different versions for long periods of time.

Users may not want to upgrade to the new version.

Bug fixes have to be released for all the versions!

Versioned model of software lifespan



Examples from Microsoft

Windows version	XP	Vista	7	8
Release date	31 Dec **2001**	30 Jan **2007**	22 Oct **2009**	30 Oct **2012**
Retail sales end	30 Jun **2008**	22 Oct **2010**	TBD	TBD
Preinstalled sales end	22 Oct **2010**	22 Oct **2011**	TBD	TBD
Mainstream support ends	14 Apr **2009**	10 Apr **2012**	12 Jan **2015**	9 Jan **2018**
Extended support ends	8 Apr **2014**	11 Apr **2017**	14 Jan **2020**	10 Jan **2023**

Other companies (e.g. Apple) are much more aggressive in stopping support (Apple usually maintains two versions of the OS at the same time)

Conclusions

Software systems have a long and eventful life

Building and evolving systems is a challenging task, that lead to the creation of many software processes.

The waterfall process has been mostly superceded by more iterative processes dealing with some aspects of the evolution.

Prototyping and early feedback are necessary!

The end of a system’s life must be managed carefully

Software systems are still serviced for a period of time, before being phased out and closed down.

The user’s data needs to be exported.

Several versions of a system can coexist in various stages of the evolution.