2.6.1 Exkurs zum Versionsverwaltungssystem GIT

GIT ist ein verteiltes Versionsverwaltungssystem. Ein wesentlicher Unterschied zwischen GIT und SVN besteht darin, dass GIT *dezentralisiert* ist. Das bedeutet, dass es keinen einzelnen zentralen Server gibt, auf dem die Gambas-Quelltexte und vor allem deren Geschichte *im Original* hinterlegt sind. Während SVN zentralisiert ist, hat bei GIT jeder Entwickler die komplette Versionsgeschichte von Gambas lokal auf seiner Festplatte. Wenn dem SVN-Server bei Sourceforge irgendwann einmal etwas zugestoßen wäre, wäre die komplette Historie der Gambas-Quelltexte verloren. Bei SVN hat jeder nur die aktuelle Version der Quelltexte – deren Geschichte ist auf einem einzigen Server gespeichert.

Das führt u.a. dazu, dass man mit GIT offline arbeiten kann. Bei SVN braucht man eine aktive Internetverbindung, um Änderungen zu 'committen' ('einchecken'). Ja, man braucht sogar eine Internetverbindung, um sich zum Beispiel die Commit-Logs der letzten Woche anzeigen zu lassen, denn die hat nur der (eine) Server. Das ist für Entwickler nervig, weil es lang dauert. GIT hat dieses Problem nicht, es wird deshalb auch als das überlegene System betrachtet.

Ein weiteres Plus von GIT aus Sicht der Entwickler sind die so genannten Branches, die Zweige der Versionsgeschichte repräsentieren. In GIT wie auch in SVN ist es möglich, eine nicht-lineare, baumartige Versionsgeschichte zu haben. Zum Beispiel könnte Boege an einer neuen Komponente basteln, während Minisini gleichzeitig Fehler korrigiert, Bodard an Beispielprojekten schraubt, Prokopowicz eine Komponente umschreibt und Lenngren vielleicht mal über einen längeren Zeitraum versucht, gb.jit auf eine neue Version von LLVM umzustellen. Das ergibt fünf Entwicklungszweige, die parallel verlaufen und sich von einem gemeinsamen Zeitpunkt in der (Gambas-)Versionsgeschichte trennen. Später wird man diese Fäden wieder zum Hauptzweig von Gambas zusammenführen – aber erst dann, wenn die Aufgabe des jeweiligen Zweigs bearbeitet und abgeschlossen ist.

Tags dagegen sind stabile Versionen und somit sind Tags keine Entwicklungszweige (→ Head losgelöst) von Gambas. Ein Tag ist kein Entwicklungszweig, von dem aus sich Gambas weiterentwickeln ließe. Ein Tag ist nur ein Sichtfenster in die Versionsgeschichte. Die Weiterentwicklung von v3.10.0 wird zum Beispiel irgendwann einmal v3.10.1 sein. Sie können sich zum Beispiel alle Tags (also alle stabilen Versionen von Gambas) auflisten lassen:

```
$ cd pfad_zum_gambas_repositum
$ git tag | sort -V
```

wobei die Liste sortiert ausgegeben wird. Die höchste Versionsnummer – aktuelle stabile Version – erscheint als letzter Eintrag in der Liste.

Das ganz oben Beschriebene wurde bisher in Gambas nie so gemacht. Es war immer so, dass alle Entwickler gleichzeitig und vollkommen nebeneinander ihre Aufgaben auf dem *einzigen* Hauptzweig der Versionsgeschichte verfolgt haben. Der Grund dafür ist, das Branches in SVN unfassbar träge Gebilde sind. Ein Branch in SVN ist im Grunde eine komplette Kopie des gesamten Repositoriums und das sind sehr viele Dateien.

Die verteilte Natur von GIT begünstigt unabhängige Entwicklungen und entsprechend wichtig sind dort auch die Branches, und die sind schlank implementiert! Einen neuen Branch zu erzeugen geht in einem Wimpernschlag, man kann sofort loslegen. Ist man dann mit dem Branch fertig, wird er (wenn nichts schief gelaufen ist) in Rekordzeit in den Hauptzweig der Versionsgeschichte eingepflegt. Letztere bleibt dadurch sehr schön geordnet und wird außerdem nicht durch halbfertige Ansätze instabiler gemacht als sie sein muss. Branches gelten als eine der größten Stärken von GIT.

Noch einmal zurück zum Aspekt Dezentralisierung: In GIT hat jeder seine eigene lokale und unabhängige Kopie von Gambas auf der Festplatte. In der Praxis werden die Entwickler sich aber austauschen und ihre Commits miteinander teilen. Das verweist auf die zweite Ebene, die oben angedeutet wurde. Auf der oberen Ebene befindet sich die Webseite GitLab.com, die ab 12. August 2017 SourceForge.-net ersetzt. Mit dem Umzug der Mailingliste am 15. Oktober 2017 nach http://lists.gambas-basic.org. ist der Wechsel von SVN auf GIT abgeschlossen. GitLab ist ein Host für GIT-Projekte. Dort kommen die Entwickler eines Projekts zusammen und tauschen ihre eigenen Änderungen der Software miteinander aus. Man kann dort zum Beispiel auch Fehler melden oder den Entwicklern Quelltext zusenden, den diese dann prüfen und in das Projekt übernehmen können.

Obwohl GitLab wie der zentrale GIT-Server des Gambas-Projekts aussieht, ist die Situation anders als bei SVN und SourceForge. Wie bereits betont, hat bei GIT jeder Entwickler die gesamte Versionsgeschichte von Gambas und auch Sie, wenn Sie sich mit *$ git clone https://gitlab.com/gambas/gambas* das Repositorium von Gambas auf den heimischen PC in den Standard-Ordner 'gambas' kopiert haben. Die Versionsgeschichte von Gambas ist also sicher, hängt nicht von der Zuverlässigkeit eines einzigen Servers auf SourceForge wie bei SVN ab. Man kann sich außerdem als Entwickler entscheiden, ob und wann man seine Commits veröffentlicht. In SVN war es Bestandteil des Commit-Prozesses, dass die Änderungen sofort hoch geladen und veröffentlicht wurden. Bei GIT kann man in aller Ruhe eine ganze Reihe von Commits und Tests durchführen und später alles zusammen sammeln, u.U. sogar die Commit-Historie ändern und dann hoch laden.

Zukünftig wird GitLab die zentrale Anlaufstelle sein, um die Gambas-Quellpakete zu bekommen. Das ist auch richtig so, denn durch das unterliegende System GIT sind Sie vor einigen Gefahren der Zentralisierung gefeit. Auch die Arbeit für Entwickler wird in einigen Punkten angenehmer.

Bleibt die Frage, was sich für die Entwickler und Nutzer beim Umstieg von SVN auf GIT ändert? Es ändert sich nur der Zugang, wie man an die Quelltexte kommt. Alles andere, wie zum Beispiel die Installation von Gambas, bleibt gleich. Im Grunde müssen Sie nur die *zwei* svn-Kommandos ersetzen, die man früher benutzt hat:

SVN – SourceForge:

```
# Das ganze Repositorium laden
$ svn checkout svn://svn.code.sf.net/p/gambas/code/gambas/trunk
# Updates holen
$ svn update
```

GIT – GitLab:

```
# Das ganze Repositorium laden
$ git clone https://gitlab.com/gambas/gambas
# Updates holen
$ git pull
```

Weitere Hinweise zu GIT finden Sie u.a. auf diesen Seiten:

```
http://gambaswiki.org/wiki/howto/git ' Pflichtlektüre
https://gitlab.com/gambas/gambas
```

```
https://git-scm.com/book/en/v2
https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet
https://de.wikipedia.org/wiki/Git
https://rogerdudler.github.io/git-guide/
https://www.sbf5.com/~cduan/technical/git/
```