Ein Blick voraus

des Autors von C++: Bjarne Stroustrup

Inhalt

- Einleitung
- Rückblick
- Nur eine Übergangslösung?
- Was würde C++ effektiver machen?
- Quelle

Einleitung

- Wo steht C++, und wo könnte es stehen?
- Künftige Probleme für Programmentwicklung und Programmiersprachen und wie kann C++ sie behandeln?
- Auf welchen Gebieten kann der Nutzen von C++ deutlich verbessert werden?

Rückblick

- 3 schwierige Fragen
  - Hat C++ seine Bestimmung erfüllt?
  - Ist C++ eine kohärente Sprache?
  - Was war der größte Fehler?
- Antworten
  - Ja, ja, die Bibliothek von Version 1.0

Hat C++ seine Bestimmung erfüllt?

- C++ ist eine Mehrzweckprogrammiersprache, die das Programmieren angenehmer machen soll → dabei erfolgreich
- Programme auf einer höheren Abstraktions-ebene zu schreiben, ohne Effizienz einzubüßen → ja, bei komplexeren Anwendungen

Hat C++ seine Bestimmung erfüllt?

- C++ machte Objektorientierung und Daten-abstraktion zugänglich
- Dazu trugen 3 Dinge bei
  - Erzeugt Code der mit Laufzeit- und Platzeigen-schaften gegenüber C mithalten kann
  - Der Code kann integriert werden
  - C++ ermöglicht einen fließenden Übergang zu den neuen Programmiertechniken

Ist C++ eine kohärente Sprache?

- Was sollte und könnte anders sein?
- Grundsätzliche Entscheidungen
  - Statische Typprüfung
  - Trennung von Sprache und Umgebung
  - Verträglichkeit mit C
  - Kein verlassen auf garbage collection

Ist C++ eine kohärente Sprache?

- Was hätte weggelassen werden können?
- Gibt keinen Teil der weggelassen werden kann, ohne auf wichtige Methoden zu verzichten
- Der Grund: C++ vereint 3 Sprachen in einer
  - Programmierung auf niedriger Ebene (wie C)
  - Techniken für abstrakte Datentypen (wie Ada)
  - Objektorientierte Programmierung (wie Simula)

Ist C++ eine kohärente Sprache?

- Was hätte hinzugefügt werden können?
- Sowenig wie möglich hinzufügen
- Aus einem Brief im Namen des C++ Komitees
  - Erst versuchen die Erweiterung auszureden
  - Änderungen sind genau abzuwägen
  - Zusätze werden mit großer Angst unternommen
  - Bevorzugen Alternativen
  - Könnte C++ inkohärent machen, anstatt besser

Was war der größte Fehler?

- Version 1.0
- Fehlen fundamentaler Klassen in der Bibliothek.
- Jeder erfand sie neu → falscher Aufwand

Nur eine Übergangslösung?

- traditionelle Programmierung im Vergleich zu objektorientierter Programmierung, Datenabstraktion
- Eine Sprache existiert solange es Probleme gibt, die sie zu lösen hilft und es keine andere gibt, die eine viel bessere Lösung bietet
- Bleiben Probleme, die C++ hilft zu lösen, bestehen?
- Werden wesentlich bessere Lösungen aufkommen?
- Wird C++ gute Lösungen für neue Probleme liefern?
- Viele werden, langsam, ja

Nur eine Übergangslösung?

- Der Übergang hinzu C++ wird in 5 Jahren nicht abgeschlossen sein
- C++ für hybrid Design und Entwicklung wird das Jahrhundert überdauern
- Und C++ zur Pflege und Erweiterung alten Quelltextes wird länger noch bestehen

Nur eine Übergangslösung?

- Wenn C++ die Antwort ist, was ist die Frage?
  - C++ ist eine Mehrzwecksprache
  - Eine spezifische Frage → eine spezielle Sprache
  - Vorteil von C++: gute Antwort auf viele Fragen
- Ein Gebiet, wo jemand keine bessere Sprache als C++ entwickeln kann, ist kaum vorstellbar
  - Gebiete wo C++ grundsätzlich Stärken hat

Nur eine Übergangslösung?

- Systemprogrammierung auf niedriger Ebene
- Systemprogrammierung auf höherer Ebene
- Eingebetteter Quelltext
- Numerisches, wissenschaftlichen Berechen
- Normale Anwendungsprogrammierung
- Vermischte Systeme

Was würde C++ effektiver machen?

- Erwartungen wurden selbst von den fantastischsten Verbesserungen übertroffen
- Vieles kann dazu beitragen C++ Implementierungen nützlicher zu machen und vieles können Programmierer und Designer lernen, um selbst effektiver zu werden
- Was könnte C++ Programmierung effektiver machen?

Was würde C++ effektiver machen?

- Stabilität und Standards
  - Stabilität der Sprachdefinition ist die Grundlage für die weitere Fortentwicklung
  - Der Standard unterstützt C++ und andere Organisationen arbeiten an neuen

Was würde C++ effektiver machen?

- Ausbildung und Techniken
  - Große Verbesserungen durch einfaches Lernen von Design- und Programmiertechniken
  - Einfachster und billigster Weg C++ effektiver zu nutzen
  - Jedoch auch schwer Gewohnheiten zu ändern

Was würde C++ effektiver machen?

- Aufgaben des Systems
  - Dynamisches Verlinken
  - Unabhängigsein der Bibliothek von seiner Implementierung
- Hinter Dateien und Syntax
  - Schrittweise Kompilierung
  - Einfache Fragen sofort beantworten

Zusammenfassung

- C++ Stärke liegt in vielen Dingen, nicht nur in einer Sache
- Ein Fortschritt ist nur durch Verbesserung von viele Dingen möglich

- Wie sieht es nun aus, über 10 Jahre danach?

Quelle

- Bjarne Stroustrup,
  The Design and Evolution of C++,
  Addison-Wesley 1994