UNIVERSIDAD NACIONAL
DEL ALTIPLANO
"FACULTAD DE INGENIERÍA
ESTADÍSTICA E
INFORMÁTICA"

The Software Crisis

Learners:
Luz Bella Valenzuela Narvaez
Nelyda Ayde Condori Condori
Lindell Dennis Vilca Mamani
Lisbet Yamira Quenta Anco
Roger Ausberto Mamani Ramos

Teacher:

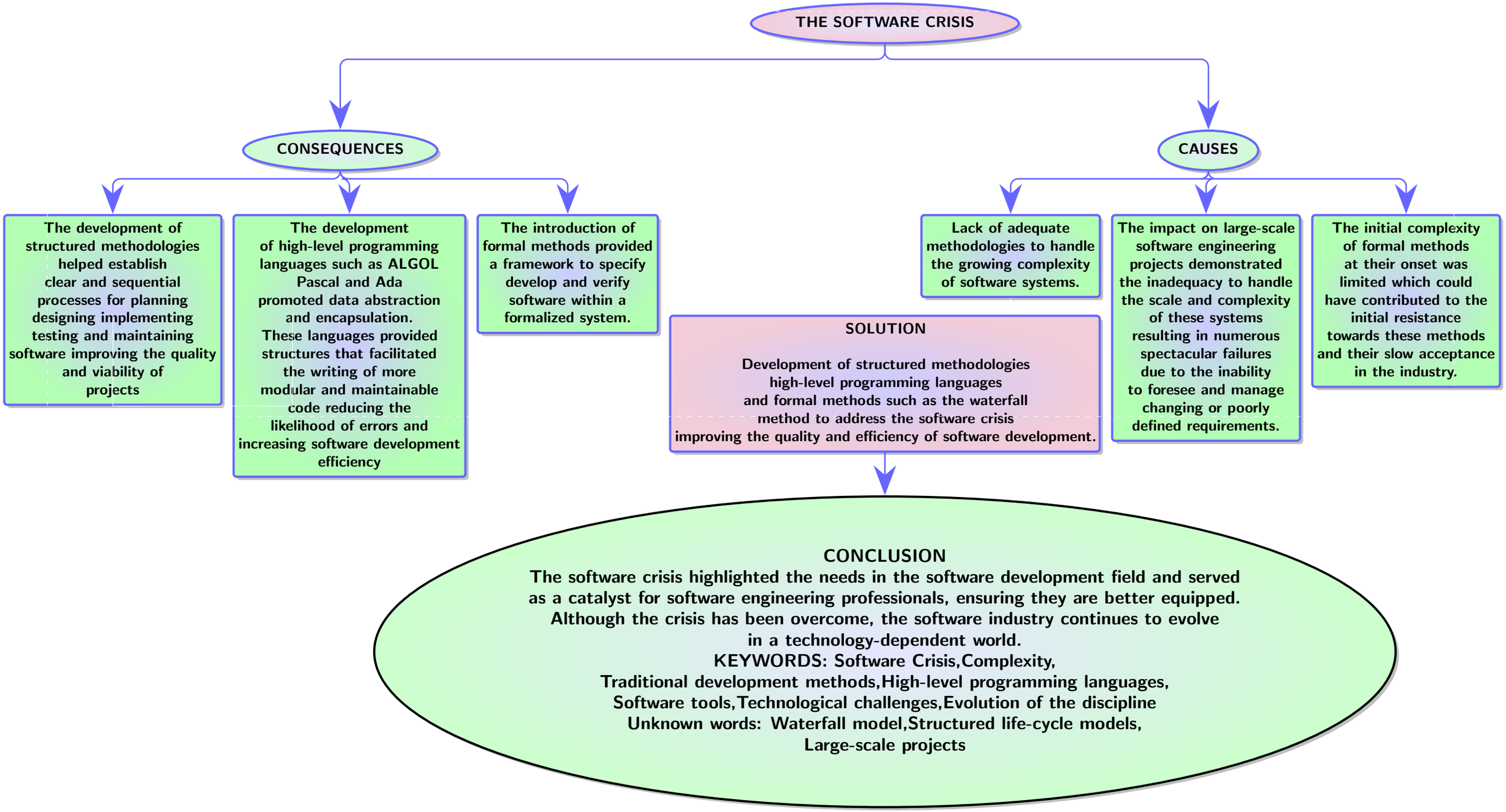
TORRES CRUZ FRED

Course:

SOFTWARE ENGINEERING

Semester VII

PUNO, PERÚ
2024


THE SOFTWARE CRISIS
CONSEQUENCES
CAUSES
The development of structured methodologies helped establish clear and sequential processes for planning designing implementing testing and maintaining software improving the quality and viability of projects
The development of high-level programming languages such as ALGOL Pascal and Ada promoted data abstraction and encapsulation. These languages provided structures that facilitated the writing of more modular and maintainable code reducing the likelihood of errors and increasing software development efficiency
The introduction of formal methods provided a framework to specify develop and verify software within a formalized system.
Lack of adequate methodologies to handle the growing complexity of software systems.
The impact on large-scale software engineering projects demonstrated the inadequacy to handle the scale and complexity of these systems resulting in numerous spectacular failures due to the inability to foresee and manage changing or poorly defined requirements.
The initial complexity of formal methods at their onset was limited which could have contributed to the initial resistance towards these methods and their slow acceptance in the industry.
SOLUTION
Development of structured methodologies high-level programming languages and formal methods such as the waterfall method to address the software crisis improving the quality and efficiency of software development.
CONCLUSION
The software crisis highlighted the needs in the software development field and served as a catalyst for software engineering professionals, ensuring they are better equipped. Although the crisis has been overcome, the software industry continues to evolve in a technology-dependent world.
KEYWORDS: Software Crisis,Complexity, Traditional development methods,High-level programming languages, Software tools,Technological challenges,Evolution of the discipline
Unknown words: Waterfall model,Structured life-cycle models, Large-scale projects