**Slide 1: COSMIC Full Function Points (FFP) – Title Slide**

**Speaker Notes:**

**Good [morning/afternoon], everyone. I’m Mohammad Alquamah Ansari, and today I’ll be presenting a rigorous and scientific method for measuring software size—COSMIC Full Function Points, or COSMIC FFP.**

**This presentation is grounded in international standards—ISO/IEC 19761—and real-world academic case studies. Our goal is to understand how COSMIC fills a crucial gap in software engineering: establishing a universal, objective unit for measuring software functionality.**

**Slide 2: The Core Problem – The Absence of a Unit**

**Speaker Notes:**

**The first principle in any scientific discipline is unit standardization—meters in physics, kilograms in mass. However, software engineering historically lacks a standard unit of size.**

**Let me illustrate:**

* **Lines of Code (LOC) measure the *solution*, not the *problem*. A shorter or longer codebase doesn’t necessarily imply more or less functionality.**
* **Staff-months measure *effort*, not *size*. For example, the same size software can take different time depending on team skills.**

**Without a standard unit, project estimation, benchmarking, and productivity tracking remain guesswork, not engineering.**

**Slide 3: The COSMIC Meta-Model & Core Principles**

**Speaker Notes:**

**COSMIC stands for Common Software Measurement International Consortium. It introduces a universal model to measure the *functional size*—what the software does for the user, not how it's coded.**

**Key concepts:**

* **Triggering Event: Every function begins with an external stimulus—like a user click or sensor reading.**
* **Functional User: The source/destination of interaction—human or external system.**
* **Data Group: Logical group of data—like customer profile or invoice.**
* **Software Boundary: Think of this as a firewall between system and environment. We only measure interactions that cross this boundary.**

**For example, a banking app that retrieves and shows a balance has user-triggered interaction, and data moves in and out across this boundary.**

**Slide 4: The Four Data Movements – The Quantum of Work**

**Speaker Notes:**

**In COSMIC, each functional movement of data is atomic and measurable—each is worth 1 CFP (COSMIC Function Point).**

**The four movements are:**

1. **Entry (E) – Data flows from user to system (e.g., submitting a form).**
2. **Exit (X) – Data flows from system to user (e.g., displaying confirmation).**
3. **Read (R) – Data fetched from storage to system (e.g., fetching user profile).**
4. **Write (W) – Data sent from system to storage (e.g., storing new password).**

**Each of these movements represents a unit of functional work, independent of technology or implementation. This makes the approach language-agnostic and architecture-neutral.**

**Slide 5: The Measurement Process – A Stepwise Formalism**

**Speaker Notes:**

**COSMIC provides a step-by-step methodology, ideal for scientific repeatability and auditability:**

1. **Define the Scope: Clarify what will be measured—modules, APIs, or full applications.**
2. **Identify Functional Processes: Each user event—like "place order" or "view report"—is a process.**
3. **Identify Data Movements: Within each process, count E, X, R, and W movements.**
4. **Calculate Total FFP: Add up all data movements. Simple, transparent, and repeatable.**

**This method brings discipline into software estimation, similar to metrology in engineering.**

**Slide 6: Case Study – Hotel Reservation System**

**Speaker Notes:**

**Let’s walk through a realistic case study: a hotel reservation system.**

**Functional Process: A guest books a room.**

* **Entry (1): Guest inputs reservation details → system.**
* **Read (1): System checks room availability from the DB.**
* **Write (1): System stores booking.**
* **Exit (1): Confirmation message is displayed to the guest.**

**Total = 4 CFP.  
This simple example illustrates how COSMIC can model functional complexity *without a single line of code*. It works for both business software and embedded systems.**

**Slide 7: Advantages of COSMIC FFP**

**Speaker Notes:**

**Why adopt COSMIC FFP over older methods?**

**✅ Technology-Independent: Doesn’t care whether you’re using Python, Java, or no code at all.  
✅ Cross-Domain Usability: Works for mobile apps, business systems, embedded controllers.  
✅ Supports Benchmarking: Lets us compare different vendors or versions objectively.  
✅ Reliable for Outsourcing: Governments and industries use it for *contractual payment by functionality*, not time or code volume.**

**Imagine two teams building the same banking module. COSMIC ensures they’re paid for delivered functionality, not code bloat.**

**Slide 8: Disadvantages & Limitations**

**Speaker Notes:**

**No method is perfect. COSMIC comes with:**

**❌ Steep Learning Curve: Analysts need proper training.  
❌ Requires Clear Specs: Vague requirements lead to incorrect measurements.  
❌ Time-Consuming: Manual data movement analysis can be slow for large systems.  
❌ Limited Tooling: Fewer tools support COSMIC than traditional function point analysis.**

**These drawbacks are addressable through automation, improved tooling, and training. As academia and industry adopt COSMIC more widely, these challenges are decreasing.**

**Slide 9: From Size to Effort – The Foundation of Estimation**

**Speaker Notes:**

**COSMIC’s true power shines when we link size to effort.**

**Using the equation:  
Effort = Productivity × Size**

* **COSMIC provides Size in FFP.**
* **Productivity can be calibrated from past projects (e.g., 5 FFP/day per developer).**

**This creates a data-driven model for project planning, reducing bias and overestimation.  
Imagine planning a national healthcare system—COSMIC lets you estimate with scientific precision.**

**Slide 10: Conclusion – Key Takeaways**

**Speaker Notes:**

**In conclusion:**

* **COSMIC FFP provides a formal, auditable, and objective way to measure software functional size.**
* **It addresses fundamental gaps in software engineering by offering a unit of measurement, like meters or grams.**
* **Despite its learning curve, the payoff is immense: better estimation, fairer outsourcing, and transparent benchmarking.**

**As future AI engineers and ethical technologists, we must embrace rigorous, scientific practices like COSMIC to build reliable, fair, and scalable systems.**