Nesneye Yönelik Yazılım Mühendisliği (376)

Dr. Öğr. Üyesi Ahmet Arif AYDIN

3.Hafta

Yazılım Mühendisliğine Giriş

Yazılım Mühendisliğine Giriş

Software (Yazılım) nedir ?

- Bilgisayar **programları, prosedürleri** ve bir bilgisayar sisteminin işletilmesine ilişkin **dokümantasyon** ve **veriler** yazılım olarak adlandırılır (*Computer **programs**, **procedures**, and possibly associated **documentation** and **data** pertaining to the operation of a computer system*)

- Bir yazılım varlığı **veri kümeleri** (data sets), **algoritmalar(**algorithms**)**, **fonksiyonlar (**functions**)** ve **aralarındaki ilişkilerden** (relations) oluşur.

Yazılım Mühendisliğine Giriş

Yazılım mühendisliği (Software Engineering)

- Yazılımın **geliştirilmesi** (development), **işletilmesi-çalıştırılması** (operation) ve **bakımı** (maintenance) aşamalarının sistematik, disiplinli ve ölçülebilir bir yaklaşımın uygulanması ve bu alanlardaki çalışmalar yazılım mühendisliği olarak bilinir.

- The application of a **systematic**, **disciplined**, **quantifiable** approach to the ***development***, ***operation***, and ***maintenance*** of software

IEEE: Institute of Electrical and Electronics Engineers, 1963

"IEEE Standard Glossary of Software Engineering Terminology," Office, vol. 121990, no. 1, p. 1, 1990.

No Silver Bullet ! (Yazılım Problemleri)

1. **Hatalı yönetim** *(wrong software management)*
2. ***K*açırılan teslim tarihi** *(missed schedules & deadlines)*
3. ***B*ütçe ve ödeme problemleri** *(payment & budget problems)*
4. ***H*atalı - kusurlu ürünler** *(flawed products)*

Bahsedilen problemler yazılım projelerini canavarlara dönüştürebilir!

Bu canavarları ortadan kaldırmak için bir sihirli değnek yok !

(**NO SİLVER BULLET**) !

Frederic P. Brooks,No Silver Bullet: Essence and Accident of Software Engineering , 1987

No Silver Bullet ! (Yazılım Problemlerinin Çözümü)

Yazılım problemlerini ortadan kaldırmak için yapılması gerekenler

1. **Mizah ve şeytani teorilerini faydalı gelişim teorisiyle değiştirmek** (replacing the demon and humors theories by the **germ theory)**
2. Aşamalı olarak ilerleme (s**tepwise progress**)
3. **Persistent** (*ısrarcı*) & **unremitting** (*aralıksız*) effort

Frederic P. Brooks,No Silver Bullet: Essence and Accident of Software Engineering , 1987

Yazılım Teknolojilerinin zorlukları

Fred Brooks yazılım teknolojilerinin zorluklarını iki ana kategori altında tanımlamaktadır. (*Major difficulties of software technology can be divided in Essence and Accidents*).

Öz ile alakalı (**Essence**) olan ve görmezden gelinemeyen zorluklar (***Essence*** is *irreducible challenges that inherited from the nature of software)*

1. Complexity
2. Conformity
3. Changeability
4. Invisibility

Yazılım Teknolojilerinin zorlukları

Fred Brooks yazılım teknolojilerinin zorluklarını iki ana kategori altında tanımlamaktadır. (*Major difficulties of software technology can be divided in Essence and Accidents*).

Öz ile alakalı (**Essence**) olan ve görmezden gelinemeyen zorluklar (***Essence*** is *irreducible challenges that inherited from the nature of software)*

1. Complexity
2. Conformity
3. Changeability
4. Invisibility

Ürün ve alan ile alakalı zorluklar (Accidents)

Accidents *are product related and domain specific problems*

Frederic P. Brooks,No Silver Bullet: Essence and Accident of Software Engineering , 1987

Essence (Öz ile Alakalı) Zorluklar (Complexity)

1. *Karmaşıklık (Complexity)*

- Projenin boyutuna bağlı olarak liner olmayan bir biçimde artar.(*increases in nonlinear fashion when their size scale-up)*

- Birbirine benzeyen iki parça bulunmamaktadır *(there is no alike two parts: variety of modules and classes)*

- Diğer akademik alanlarda bu özellik göz ardı edilebilir *(In math and physics complex phenomena's are modeled by simplified models with "ignoring complexities", however, complexity can't be ignored in software since it is an "essential feature")*

Frederic P. Brooks,No Silver Bullet: Essence and Accident of Software Engineering , 1987

Essence (Öz ile Alakalı) Zorluklar: Conformity

2. *Uyum (Conformity)*

- Yazılım dış dünyadan izole edilmiş bir biçimde geliştirilemez. Geliştirilen yazılım dış dünyada bulunan kısıtlamalarla ve standartlara uyumlu olmak zorundadır. **(***A software entity can't be produced in isolation, it must compatible (conform) to real-world constraints)*

- Kısıtlamalar mevcut olan donanım önceden var olan donanım, üçüncü taraf bileşenleri, devlet düzenlemeleri ve eski veri formatları **(***such as pre-existing hardware, third party components, government regulations, and legacy data formats)*

Essence (Öz ile Alakalı) Zorluklar: Changeability

3. *Changeability (Değişebilirlik)*

- *Yazılım sürekli değişen bir ortam içerisinde geliştirilir. (Software entities are built on "**constantly changing environments**" applications, users, laws, hardware*)
- Yazılım değişim isteklerine karşı esnek olmalıdır (*Software entity must be resilient for change requests which can be user request for new features or compatibility need for new hardware*)
- Yazılım fikir tabanlı bir yapı olduğu için değişimler araba, bina gibi yapılara göre daha kolaydır. (*Since software is **pure thought-stuff**, therefore it can be changed more easily than buildings, car, etc)*

Essence (Öz ile Alakalı) Zorluklar : invisibility

4. Invisibility (Görünmezlik)

- **Yazılım somut bir varlık değildir (***Software is invisible (un-visualizable) because the reality of software is not embedded in space. Silicon chips have diagrams, computers have connectivity diagrams, and however, there is not a solid diagram for software***)**

- **Yazılımı somut olarak modellenmeye çalışılmaktadır** *(we can't see a tangible, we create abstractions, and create simple models)*

Frederic P. Brooks,No Silver Bullet: Essence and Accident of Software Engineering , 1987

Essence (Öz ile Alakalı) Zorluklar : invisibility

4. Invisibility (Görünmezlik)

- **Veri ve kontrol akışı bağımlılık kalıpları kolaylıkla tek bir şema yardımıyla temsil edilememektedir** (*Data flow, Control flow, dependency patterns are not easily represented in one representation)*

- **Yazılım sistemlerinin yapısı tasarlanırken ortak çalışma gereklidir**

  (collaboration is necessary for designing structure of a software system)

Frederic P. Brooks,No Silver Bullet: Essence and Accident of Software Engineering , 1987

Essence (Öz ile Alakalı) Zorluklar

Öz ile alakalı (**Essence**) olan ve görmezden gelinemeyen zorluklar (***Essence*** is *irreducible challenges that inherited from the nature of software)*

Complexity (Karmaşıklık)	**Conformity** (Uyum)
Changeability (Değişebilirlik)	**Invisibility** (Görünmezlik)

ACCIDENTS:

Ürün (product) ve ***alan (domain)*** ile alakalı zorluklar

Özel bir alan ve proje ile alakalı olarak ortaya çıkabilen problemler *(product related , domain specific problems)*

Frederic P. Brooks,No Silver Bullet: Essence and Accident of Software Engineering , 1987

Accidents (Ürün ile Alakalı)

Özel bir alan ve proje ile alakalı olarak *(product related , domain specific problems)* ortaya çıkabilen problemler aşağıda kullanılan yöntem ve tekniklerle desteklenmiş ve bu alanlardaki gelişmeler günümüzde de devam etmektedir.

1. High-level languages (yüksek seviyeli diller)
2. Time Sharing (İşletim sisteminin(CPU) zaman paylaşımı)
3. Unified programming environments (birleştirilmiş programlama ortamları)

Frederic P. Brooks,No Silver Bullet: Essence and Accident of Software Engineering , 1987

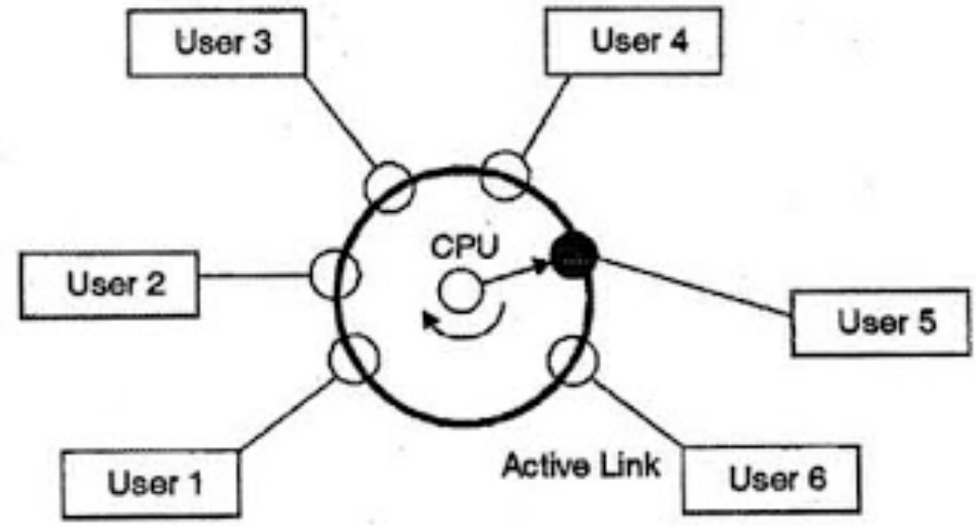
Accidents (Ürün ile Alakalı)

1. **Yüksek Seviyeli Diller (high-level languages)**
   - Makine dili ile bit seviyesinde işlem gerekleştirilirken yüksek seviyeli diller veri tipleri, işlemler gibi soyut yapıları kullanarak problemin çözümünü sağlar. (*Concrete machine program focused on bits, registers, conditions, branches and channels high-level (abstract) programs consist of conceptual constructs such as operations, data types, sequences and communication)*
   - Yüksek seviyeli diller düşük seviyeli dillerin karmaşıklığını azaltmıştır (*High level languages reduced whole low level code complexity with abstract & conceptual constructs)*
   - *productivity, simplicity, comprehensibility and reliability*

Frederic P. Brooks,No Silver Bullet: Essence and Accident of Software Engineering , 1987

Accidents (Ürün ile Alakalı)

1. **High-level languages**
2. **Time sharing**
   - Zaman paylaşımı
   - 1960'lı yıllarda ortaya çıkmıştır
   - Programcıların ortak ve aynı anda kaynakları kullanmasını ve çalışmasını sağladığından daha yüksek kalite ve daha hızlı bir biçimdeki üretimi desteklemektedir.
   - 'the distribution of a computing resource to many users via ***multiprogramming*** or **multitasking'** (https://www.techopedia.com/definition/9731/time-sharing)

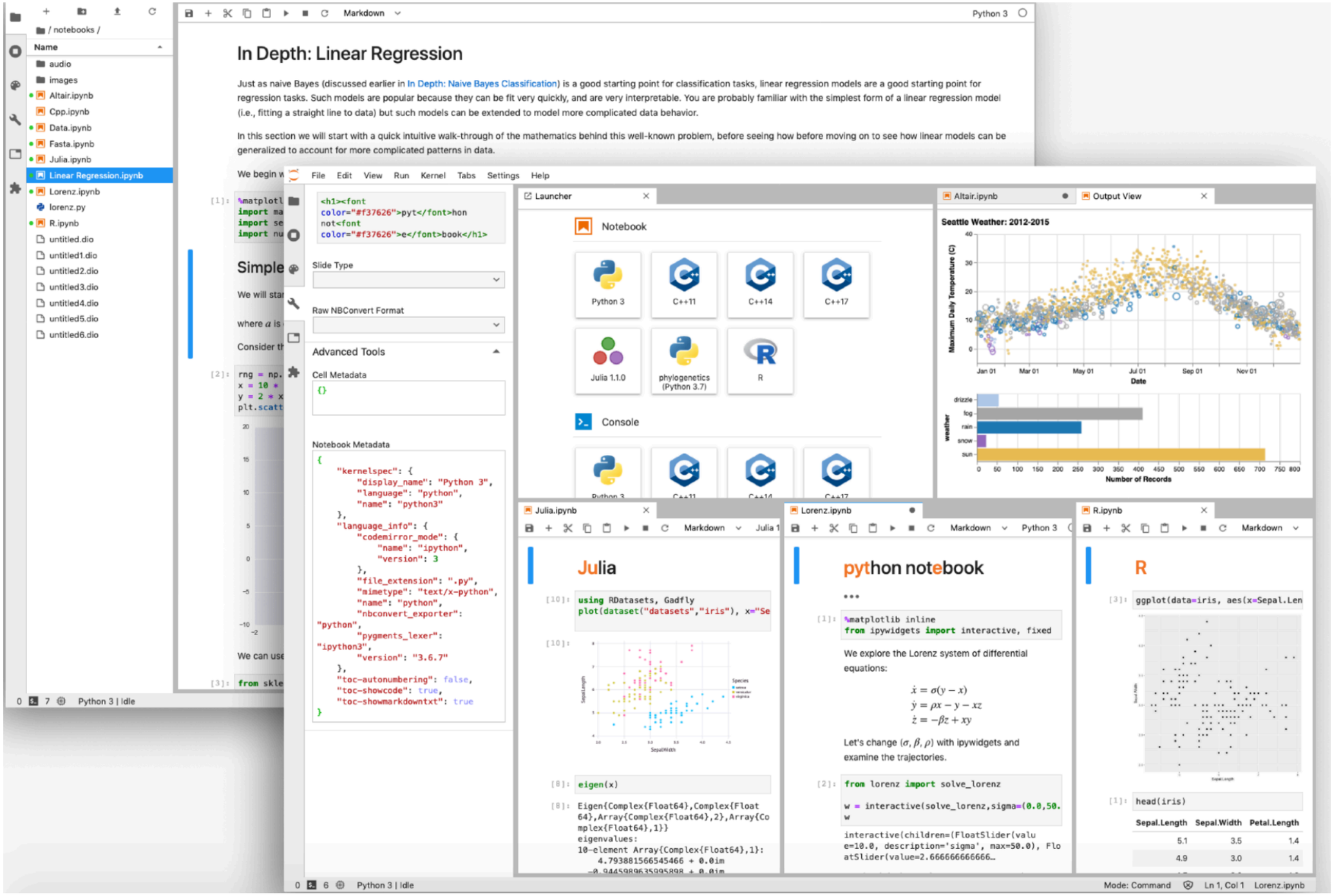


Accidents (Ürün ile Alakalı)

1. **High-level languages**
2. **Time sharing**
3. **Unified programming environments (**Birleştirilmiş programlama ortamları**)**
   - İlk defa **Unix and Interlisp** ilk birleştirilmiş programlama ortamıdır.
   - Günümüzde ise popüler olarak kullanılmaktadır. (Birleştirilmiş kütüphaneler, dosya formatları)

Accidents (Ürün ile Alakalı)

Unified programming environments (Birleştirilmiş programlama ortamları)



https://jupyter.org/