Model Bağlam Protokolü (MCP) Hakkında Kapsamlı Literatür İncelemesi

Bölüm 1: MCP'ye Giriş (Sentez)

1.1 Ajans Paradigması Değişimi ve Entegrasyon Zorluğu

Büyük Dil Modellerinin (LLM) evrimi, yapay zeka alanında temel bir paradigma değişikliğini temsil eder ve modelleri pasif metin üretiminin ötesine, gerçek dünyadaki görevleri yerine getirebilen aktif, otonom bir ajansa doğru taşır. Bu ajans dönüşümü, harici araçların çağrılması için sağlam ve ölçeklenebilir mekanizmalar gerektirir. [1] Tarihsel olarak, LLM'lerin harici yeteneklerle entegrasyonu, **entegrasyon zorluğu** nedeniyle engellenmiştir. Bu senaryoda, LLM platformları, mevcut harici araçların veya API'lerin her biri için özel, sabit kodlu bağlamalar gerektiriyordu. Bu da farklı entegrasyon yollarına yol açarak aşırı bakım maliyetlerine, yinelemelere ve ekosistem ölçeklendirilmesinde önemli engellere neden oluyordu.[2]

Model Bağlam Protokolü (MCP), bu entegrasyon darboğazını çözmek için özel olarak geliştirilmiştir. Protokol, çerçeveye özgü, uygulama merkezli araç bağlamalarından, birleştirilebilir ve dinamik olarak keşfedilebilir ağ hizmetlerinden oluşan, birlikte çalışabilir bir ekosisteme geçişi öngörmektedir.[2] LLM ile dış dünya arasındaki arayüzü standartlaştırarak, MCP yinelenen bakım çabalarını ortadan kaldırır ve araçlarla desteklenen yapay zeka için paylaşımlı, ölçeklenebilir bir ekosistem oluşturur.[2]

1.2 Model Bağlam Protokolünü (MCP) Tanımlama: "Evrensel Bağlayıcı"

Anthropic tarafından 2024 yılının sonlarında tanıtılan Model Context Protocol (MCP), AI sistemlerinin temel model sınırlarının dışındaki harici verilere, API'lere ve araçlara erişmesi için

tutarlı bir mekanizma sağlayan, açık kaynaklı, şema odaklı bir standarttır. [1] Genellikle AI için "evrensel konektör" olarak nitelendirilen MCP, gerçek zamanlı karar verme için tasarlanmıştır ve ölçeklenebilir, akıllı ajan iş akışları oluşturmanın temelini oluşturur.[3]

MCP'nin mimari uygulaması, çekirdek LLM akıl yürütme alanı (istemci) ile aracın yürütme ortamı (sunucu) arasında katı bir ayrım uyguladığı için önemlidir.[4] Bu ayrıştırma, mimari esnekliği ve modülerliği sağlamak için hayati önem taşır. Müşteri (ajan) veya ana bilgisayar kodunda ilgili değişiklikler yapılmasına gerek kalmadan yeni araçlar eklenebilir veya mevcut araçlar güncellenebilir, böylece LLM'ler talep üzerine yeni sunuculara bağlanarak işlevselliklerini esnek bir şekilde genişletebilirler.[2] Bu zorunlu ayrım, temel dış araç envanteri değişip büyüdükçe bile ajanın karmaşık mantığının istikrarlı kalmasını sağlar.

1.3 Kapsamlı İncelemenin Yapısı ve Kapsamı

Bu inceleme, 2024 yılından sonra yayınlanan son akademik makalelerden elde edilen bulguları sentezlemekte ve yalnızca MCP'nin mimarisi, benimsenme dinamikleri, ampirik performansı ve ilgili araştırma zorluklarını ayrıntılı olarak ele alan akademik makalelere odaklanmaktadır. Sonraki bölümlerde, temel mimari bileşenler ayrıntılı olarak ele alınacak, uygulama otomasyonundaki atılımlar analiz edilecek, genel ve özel alanlardaki uygulama alanları incelenecek ve son olarak, ajans güvenilirliği, güvenlik ve yönetişim ile ilgili kritik çözülmemiş sorunlar tartışılacaktır.

Bölüm 2: Önemli Akademik Makaleler (Özet Listesi)

Aşağıdaki özetler, Model Bağlam Protokolü'nün geliştirilmesi, uygulanması ve ampirik değerlendirmesine odaklanarak, mevcut araştırmaları ve Model Bağlam Protokolü'nün anlaşılmasını yönlendiren temel literatürü temsil etmektedir.

1. Özet [1]: Büyük Dil Modelleri (LLM'ler), pasif metin üreticilerinden, aktif ajanlara doğru giderek daha fazla evrimleşmektedir...kaynak
2. Özet [5]: Araç çağırma, AI ajanlarının gerçek dünyayla etkileşime girmesi ve karmaşık sorunları çözmesi için kritik bir yetenek olarak ortaya çıkmıştır...kaynak
3. Özet (Bulguların Özeti) [2]: Model Bağlam Protokolü (MCP) için gelecekteki araştırma yönelimleri, standardizasyonunun, güven sınırlarının ve sürdürülebilir büyümesinin güçlendirilmesine odaklanmaktadır. Belirlenen temel zorluklar arasında güvenlik, ölçeklenebilirlik ve yönetişim sorunları bulunmaktadır. MCP sunucu yönetiminin bireysel

geliştiriciler tarafından bağımsız olarak gerçekleştirilmesi, güvenlik temelini denetleyecek veya tek tip uyumluluğu sağlayacak merkezi bir otoritenin bulunmaması anlamına gelir ve bu da tutarsız yama uygulamaları ve yapılandırma sapmalarına yol açar. Gelecekteki araştırmalar, ekosistem çapında dayanıklılığı sağlamak için güven sınırlarının güçlendirilmesi, zorunlu yapılandırma doğrulaması, otomatik sürüm kontrolleri, bütünlük denetimi ve sürekli topluluk katılımına odaklanmalıdır. Sunucu etkileşimlerinin birden fazla yaşam döngüsü aşamasına yayılması, etik ve güvenlik endişelerini artırmakta ve araç seçiminde adaleti sağlamak, veri seti sızıntılarına karşı savunma sağlamak ve otomatik karar iş akışlarında hesap verebilirliği sürdürmek için çözümler gerektirmektedir.

4. Özet [6]: LLM'lerin yetenekleri, çeşitli veri kaynaklarını veya API sonuçlarını entegre etmek için işlev çağrıları kullanılarak geliştirilir...kaynak sonuçlar.
5. Özet (Ekonomik Araştırma Uygulaması) [4]: Bu makalenin amacı, planlama, araç kullanımı vb. işlevleri yerine getiren otonom LLM tabanlı sistemler olan AI ajanlarını anlaşılır hale getirmektir...kaynak analizi.

Bölüm 3: Araştırmanın Tematik Özeti (Sentez)

3.1 Temel Tanım ve Mimari

3.1.1 Mimari Temeller: İstemci-Sunucu Modeli ve Protokol Tasarımı

Model Bağlam Protokolü, temel bir istemci-sunucu mimarisi oluşturur. MCP istemcileri genellikle, yeteneklere erişmek için sunuculara bağlanan yapay zeka ajanları veya uygulamalarıdır.[4] Bu istemciler, genişletilmiş projeler boyunca araştırma bağlamını korur ve araç kullanımı için gerekli planlamayı yürütür.[4] Tersine, MCP sunucuları kaynakları sunmak, araçları barındırmak ve harici veri kaynaklarıyla gerçek API etkileşimlerini yürütmekten sorumludur.[4]

Protokolün tasarımı, uygulama tutarlılığını ve yeniden kullanılabilirliği sağlayan JSON-RPC 2.0 standardına dayanmaktadır.[3] Bu seçim, güvenli ve otonom çalışma için gerekli özellikleri içeren, güvenlik öncelikli bir mimariyi kolaylaştırır. Bu özellikler arasında güçlü tipleme, açıkça tanımlanmış istek/yanıt yaşam döngüleri, entegre izin katmanları ve güvenli olmayan veya

istenmeyen otomatik çağırmayı önlemek için tasarlanmış istemci-sunucu alarmları için akış mekanizmaları bulunur. [3] Tanımlanmış etkileşim ve tipleme üzerine yapılan bu yapısal vurgu, güvenilirlik ve güvenliğin tartışılmaz olduğu üretim ortamlarında LLM'lerin dağıtımı için çok önemlidir.

3.1.2 Temel Bileşenler ve Şema Bağımlılığı

MCP, temel olarak, LLM tarafından dinamik keşif ve çağırma için harici araçların nasıl tanımlanacağını belirleyen şema odaklı bir standarda dayanır.[1] Harici işlevselliği tanımlayan birincil teknik bileşen, şema sözleşmesidir. Akademik literatür, bu şemaları oluşturmak ve MCP sunucularını uygulamak için en etkili yöntemin, önceden var olan endüstri standartlarını, özellikle OpenAPI 2.0/3.0 spesifikasyonlarını kullanmak olduğunu doğrulamaktadır.[1]

LLM istemcisi, aracı akıl yürütme zincirine etkili bir şekilde entegre etmek için aracın parametreleri, girdileri ve beklenen çıktıları hakkında kapsamlı bir açıklamaya ihtiyaç duyar. MCP sunucusu bu tanımları kaydeder ve LLM'nin dosya sistemleri, web tarayıcıları veya finansal veriler gibi özelliklere erişmesine olanak tanır.[6] Bu şema bağımlılığı, LLM'nin yalnızca metin üretmekten, gerçek dünya sistemlerinde çalışan geçerli işlev çağrıları üretmeye geçişi için gerekli yapıyı sağlar.

Tablo 3.1: MCP Mimari Bileşenleri ve İşlevleri

Bileşen Rolü	**Temel İşlev**	**Temel Standart/Protokol**	**Anahtar Özellik/Kısıtlama**
MCP İstemcisi (Ajan)	Araç çıktılarını keşfeder, çağırır ve LLM bağlamına entegre eder.	JSON-RPC 2.0	Bağlam penceresi sınırlamalarıyla kısıtlıdır; araç numaralandırma belirteci uzunluğunu işlemelidir.[6]
MCP Sunucusu (Araç Ana Bilgisayarı)	Dış yetenekleri (API'ler, veritabanları) ortaya çıkarır; yürütme ve kimlik	Şema odaklı (OpenAPI)	Yüksek kaliteli kaynak özellikleri gerektirir; başlangıçta manuel iskele

	doğrulamayı yönetir.		kurulumundan kaynaklanan darboğazlar vardır.[1]
Protokol Tasarımı	Araç tanımı ve etkileşimi için birleşik, standartlaştırılmış bir arayüz sağlar.	Açık Standart, JSON-RPC 2.0	Modülerlik, güvenlik öncelikli özellikler (izinler) ve ölçeklenebilir optimizasyon (önbellekleme, toplu işleme) sağlar.[3]

3.2 Uygulama, Ölçeklenebilirlik ve Benimseme Dinamikleri

3.2.1 Manuel Sunucu Geliştirme Darboğazının Nicelendirilmesi

Protokolün entegrasyonu hızlandırma hedefine rağmen, ilk benimseme araştırmaları MCP sunucularının uygulanmasında önemli sürtüşmeler olduğunu ortaya koydu. Protokolün yayınlanmasından sonraki altı ay içinde oluşturulan 22.000'den fazla MCP etiketli GitHub deposunun analizi, %5'ten azının işlevsel sunucu uygulamaları içerdiğini gösterdi. [1] Bu mevcut sunucular genellikle küçük, tek bakımcı projelerdi ve yapıştırma kodu yazma, kimlik doğrulama ve şemaları elle yapılandırma gibi tekrarlayan manuel çabalarla karakterize ediliyordu.[1] Bu, geliştiricilerin MCP'nin ortadan kaldırmayı amaçladığı entegrasyon çalışmalarının çoğunu tekrarlamasını gerektiriyordu ve yaygın benimseme için gerekli ağ etkilerini elde etmenin önündeki kritik bir başlangıç tehdidini ortaya koyuyordu.

3.2.2 Otomasyonda Çığır Açan Gelişme: AutoMCP ve OpenAPI'nin Rolü

Manuel sunucu geliştirmenin maliyetli darboğazını aşmak için, otomatik derleyici çözümleri kavramı ortaya çıktı. AutoMCP derleyici, OpenAPI 2.0/3.0 spesifikasyonlarından REST API tanımlarını ayrıştırma ve gerekli şema kaydı ve kimlik doğrulama işlemleri dahil olmak üzere

eksiksiz MCP sunucu uygulamaları oluşturma yeteneğini başarıyla gösterdi.[1]

10'dan fazla alanda 5.066 uç noktayı kapsayan 50 gerçek dünya API'sinde AutoMCP'nin ampirik değerlendirmesi önemli bir başarı sağladı. 1.023 araç çağrısından oluşan tabakalı bir örneklemden %76,5'i hazır olarak başarılı oldu.[1] Kaynak OpenAPI sözleşmelerindeki tutarsızlıklar nedeniyle gerekli olan küçük düzeltmelerin ardından ve API başına ortalama sadece 19 satırlık spesifikasyon değişikliği ile başarı oranı %99,9'a yükseldi.[1]

3.2.3 Yeni Benimseme Engeli Olarak Spesifikasyon Kalitesinin Analizi

Otomasyon analizinin sonuçları, MCP'nin benimsenmesinin teknik odağı açısından derin bir anlam taşıyor. Otomatik sunucu oluşturmanın neredeyse tamamen başarılı olması, LLM entegrasyonunun doğasında var olan karmaşıklığın temelden değiştiğini gösteriyor: Artık zorluk protokol tasarımı veya kod oluşturma değil. Bunun yerine, ekosistem genişlemesinin önündeki en büyük engel, sunucuları tanımlamak için kullanılan temel OpenAPI sözleşmelerinin kalitesi ve tutarlılığıdır. [1] Arızalar neredeyse tamamen spesifikasyon eksikliklerinden veya yanlışlıklarından kaynaklandığından, MCP ekosisteminden yararlanmak isteyen kuruluşlar API yönetişimine ve dokümantasyon doğruluğuna öncelik vermelidir. Protokolün başarısı, spesifikasyon bakımını çevreleyen eski BT uygulamaları üzerinde istemeden bir baskı noktası yaratmış ve API sözleşmelerinin yazılma ve bakımında daha fazla titizlik talep etmiştir.

3.3 Birincil Uygulama Alanları ve Gerçek Dünya Kullanım Örnekleri

3.3.1 Genel Ajan İş Akışları ve Ekosistem Büyümesi

MCP, AI tabanlı uygulamalar için temel bir mimari olarak konumunu hızla sağlamlaştırmıştır. Bu durum, GitHub ve Slack gibi yaygın kurumsal ve tüketici hizmetlerine model erişilebilir arayüzler sunan binlerce bağımsız olarak geliştirilmiş MCP sunucusu ile kanıtlanmaktadır. [2] MCPToolBench++ gibi karşılaştırmalı değerlendirmeler, 40'tan fazla farklı kategoriyi kapsayan 4.000'den fazla MCP sunucusundan oluşan bir pazardan yararlanarak, protokolün veri analizi, dosya işlemleri, finansal hesaplamalar ve genel hesaplama gibi görevler arasında geniş bir uygulama alanına sahip olduğunu doğrulamaktadır. [6] İstemci uygulamaları, gerektiğinde yeni

sunuculara bağlanarak işlevselliklerini esnek bir şekilde genişletebilir. [2]

3.3.2 Özel Alan Uygulaması: Ekonomik ve Kurumsal Araştırma

Yüksek değerli uygulama alanlarından biri, ekonomi veya kurumsal veri analizi gibi yüksek güvenilirlik gerektiren ortamlarda yapılan özel araştırmalardır. MCP, AI ajanlarının kurumsal veritabanlarına (örneğin, merkez bankaları veya özel araştırma grupları tarafından tutulan veritabanları) bağlanmasını ve bu bağlantıları sürekli olarak sürdürmesini sağlar.[4]

Bu özellik, literatür incelemelerini özerk bir şekilde yürütmek, ekonometrik kod yazmak ve hata ayıklamak, özel ekonomik verileri almak ve analiz etmek gibi karmaşık araştırma iş akışlarını mümkün kıldığı için dönüştürücü bir nitelik taşır. [4] MCP, kritik öneme sahip güçlü bir soyutlama katmanı görevi görür: karmaşık entegrasyon ayrıntılarını yöneterek, son kullanıcıların (ekonomistler veya alan uzmanları) kurumsal API ayrıntılarına ilişkin özel bilgiye ihtiyaç duymadan doğal dil veya "vibe coding" kullanarak ajanı yönetmelerine olanak tanır. [4] Entegrasyon karmaşıklığını standartlaştırılmış sunucu tasarımına aktararak, protokol, otonom araştırma iş akışları için sofistike, yüksek değerli özel veri kaynaklarının kullanımını demokratikleştirir.

3.4 Araştırma Odak Noktası: Ampirik Karşılaştırma ve Performans Analizi

3.4.1 En Son Teknoloji Benchmark'lara Giriş

MCP özellikli ajanların performansını titizlikle değerlendirmek için iki önemli kriter ortaya çıkmıştır. **LiveMCP-101**, çok sayıda farklı MCP aracının (ör. web arama, dosya işlemleri, akıl yürütme) koordineli kullanımını gerektiren, özenle seçilmiş 101 gerçek dünya sorgusu sunar. Bu yeni değerlendirme yaklaşımı, yalnızca ham API çıktılarına dayanmak yerine, gerçekçi ve dinamik yürütme senaryolarını daha iyi yansıtan temel yürütme planlarından yararlanır. [5] **MCPToolBench++**, farklı MCP sunucu yürütmelerinden gelen çeşitli yanıt biçimlerinin ve gerçek dünya ortamlarında araç başarı oranlarının doğasında var olan değişkenliğin yarattığı zorlukları ele alır. [6] Bu karşılaştırma, geniş MCP sunucu pazarı üzerine kuruludur ve hem tek

adımlı hem de çok adımlı araç çağrılarını değerlendirmek için çok alanlı bir çerçeve sunar.[6]

3.4.2 Performans Bulguları: Araç Koordinasyon Eksikliği

MCP tarafından sağlanan önemli mimari standardizasyona rağmen, performans değerlendirmeleri ajan mantığındaki temel bir sınırlamayı doğrulamaktadır. LiveMCP-101 benchmarkunu kullanan deneyler, en gelişmiş, öncü LLM'lerin bile karmaşık, çok adımlı araç düzenleme görevlerinde %60'ın altında bir başarı oranı elde ettiğini göstermektedir. [5] Bu düşük güvenilirlik oranı, MCP'nin araçlara *nasıl* erişildiğini başarıyla standartlaştırdığını, ancak bu standardizasyonun *güvenilir yürütmeyi* garanti etmek için gerekli ancak yeterli olmadığını doğrulamaktadır; temel sınırlama, dinamik ortamlarda gezinmek için gerekli olan LLM'nin uzun vadeli planlama, koordinasyon ve akıl yürütme yeteneklerinde yatmaktadır.

3.4.3 Arıza Modlarının ve Kaynak Kısıtlamalarının Niteliksel Analizi

Ayrıntılı hata analizi, MCP araçları kullanıldığında LLM'lerde belirgin arıza modları tespit etmiştir. Önemli bir kategori, "aşırı kendine güvenen kendi kendine çözümleme" olarak bilinen kritik arıza modunu da içeren Araç Koordinasyon Hatalarıdır. [5] Bu senaryoda, ajan harici bir araca ihtiyaç olduğunu fark eder, ancak harici olarak doğrulanmış ve temelli MCP aracını atlayarak kendi iç bilgisine veya muhakemesine güvenmeyi tercih eder. Bu genellikle genel veya hayal ürünü cevaplara ve görevin erken sonlandırılmasına neden olur. [5] Diğer düzenleme hataları arasında, açıkça belirtilen gereksinimleri tamamen göz ardı etmek veya ilgili aracı seçmemek sayılabilir.[5] Uygulama hataları genellikle Parametre Hataları olarak ortaya çıkar; bu durumda ajan, MCP aracı çağrısının başarılı bir şekilde yürütülmesi için gerekli olan giriş parametrelerini yanlış biçimlendirir veya atlar.[5]

Yürütme mantığındaki hataların ötesinde, araştırma LLM'nin bağlam penceresi ile ilgili kritik bir kısıtlamayı vurgulamaktadır. MCP sunucusunun çağrılması için gerekli olan araçların ve parametrelerinin metinsel açıklamaları önemli miktarda token uzunluğu tüketmektedir. [6] Ekosistem büyüdükçe (şu anda 4.000'den fazla sunucu), kullanılabilir şemaların kapsamlı bir envanterini listelemek, karmaşık planlama, akıl yürütme ve çok adımlı çıktıları işlemek için gereken kullanılabilir token alanını doğrudan azaltır.[5] Bu, envanter boyutu ve akıl yürütme derinliği arasında gerekli bir kaynak tahsisi dengesi oluşturur ve karmaşık görevlerde performansı düşürür.[5]

Tablo 3.2: MCP Etkin Ajan Yürütmede Gözlemlenen Arıza Modları (LiveMCP-101)

Hata Kategorisi	Örnek Arıza Modu	Ayrıntılı Açıklama	Kaynak
Araç Koordinasyonu	Başarı Oranı %60'ın Altında	Gerçekçi ortamlarda karmaşık, çok adımlı eylemleri koordine etmede LLM'nin başarısızlığını gösteren ampirik kanıtlar.	[5]
Araç Koordinasyonu	Aşırı Kendine Güvenen Kendi Kendine Çözüm	Ajan gereksinimi tanır ancak içsel muhakeme/bilgiyi tercih eder, bu da halüsinasyonlu çıktılara yol açar ve temelli MCP aracını atlar.	[5]
Araç Koordinasyonu	Gereksinimi Göz Ardı Etme	Ajan, açıkça belirtilen bir gereksinimi gözden kaçırır ve ilgili aracı seçmez, bu da erken sonlandırma veya genel bir nihai cevaba yol açar.	[5]
Uygulama	Parametre Hataları	Ajan, başarılı bir MCP aracı çağrısı yürütülmesi için gerekli olan giriş parametrelerini yanlış biçimlendirir veya atlar.	[5]
Ölçeklenebilirlik/Ba	Token Verimsizlikleri/Sınırl	Araç numaralandırma	[5]

ğlam	arı	(şemalar) bağlam penceresini tüketir ve karmaşık muhakemeyi tehlikeye atan bir kaynak tahsisi ödünleşmesini zorlar.	

Bölüm 4: Sonuç ve Araştırma Boşlukları (Sentez)

4.1 MCP Araştırmalarının Mevcut Durumunun Özeti

Model Context Protocol, LLM araçlarının etkileşimi için mimari temeli standartlaştırma ve önceki entegrasyon sorununu etkili bir şekilde çözme gibi temel hedeflerini gerçekleştirmiştir.[1] Araştırmalar, protokolün OpenAPI sözleşmelerine dayalı olması ve otomatik sunucu oluşturma özelliği sayesinde, geliştiricilerin AI ajanlarına harici yetenekleri sunma konusunda karşılaştıkları teknik engellerin önemli ölçüde azaldığını göstermektedir. [1] Ekosistem hızlı bir büyüme kaydetmiş, MCP'yi LLM mimarisinde önemli bir ilerleme olarak onaylamış ve ekonomik araştırma gibi yüksek riskli alanlarda sofistike uygulamaların kullanılmasını mümkün kılmıştır.[3]

Ancak, MCP tarafından gerçekleştirilen standardizasyon, acil akademik ilgi gerektiren iki kritik alanı ortaya çıkarmıştır: ajans güvenilirliğindeki temel sınırlamalar ve ekosistem yönetişimindeki derin zorluklar. [2] Topluluk, odak noktasını araç *entegrasyon* sorunlarını çözmekten, otonom *yürütme* ve *sürdürülebilirlik* sorunlarını analiz etmeye etkili bir şekilde kaydırmıştır.

4.2 Çözülmemiş Başlıca Zorluklar ve Gelecekteki Yönelimler

4.2.1 Güvenlik Açıkları ve Güçlü Güven Sınırlarının Gerekliliği

MCP'nin başarısı ve merkezi olmayan yapısı, güvenlikle ilgili sistemik riskleri beraberinde getirir. MCP sunucu yönetiminin bağımsızlığı, güvenlik temelini denetleyecek veya tek tip uyumluluğu sağlayacak merkezi bir otorite olmadığı anlamına gelir.[2] Bu merkezi olmayan yapı, dağıtım ve bakımda heterojenliği teşvik eder, tutarsız yamalama, düzensiz güvenlik uygulamaları ve zamanla değişen yapılandırmalara yol açarak güvenlik açıklarının olasılığını artırır.[2]

Gelecekteki araştırmalar, merkezi olmayan ekosistem genelinde güven sınırlarının oluşturulmasına ve güçlendirilmesine öncelik vermelidir.[2] Bu, zorunlu yapılandırma doğrulaması, otomatik sürüm kontrolleri ve bütünlük denetiminin MCP kayıtları ve koleksiyonlarına entegre edilmesi gibi teknik yönetişim çözümlerinin uygulanmasını gerektirir. Bu mekanizmalar olmadan, ekosistemin sürdürülebilir büyümesi, parçalanma ve tek tip olmayan temel uyumluluk nedeniyle yaygın güvenlik ihlalleri potansiyeli tarafından tehdit altındadır.[2]

4.2.2 Ölçeklenebilirlik Sorunları, Parçalanma ve Yönetişim Gereklilikleri

LLM'nin bağlam penceresinin katı kısıtlaması, bir ajanın tek bir çalışmada etkili bir şekilde değerlendirebileceği araçların sayısını sınırlayarak, karmaşık ve araç yoğun iş akışlarının ölçeklenebilirliğini doğrudan kısıtlamaktadır.[6] Gelecekteki araştırmalar, kapsamlı bir araç envanterini korumak ile karmaşık akıl yürütme ve planlama için yeterli token alanı ayırmak arasındaki dengeyi azaltmak için dinamik, bağlamsal araç keşif mekanizmalarına ve yenilikçi şema sıkıştırma tekniklerine öncelik vermelidir. [6]

Ayrıca, MCP, ajan etkileşiminin yüksek riskli kurumsal alanlara (örneğin, finansal modellere veya özel veri setlerine erişim) genişlemesini kolaylaştırdığından, düşük güvenilirlikle ilgili mevcut zorluklar (çok adımlı görevlerde %60'ın altında başarı oranı) daha da artmaktadır. [4] Bir ajan, koordinasyonda bir hata yaparsa veya güvenlik önlemleri yetersiz bir sunucu aracılığıyla kritik bir kaynağa erişirse, bunun sonucunda ortaya çıkan başarısızlık önemli etik, güvenlik ve yasal sonuçlar doğurur. Bu nedenle, gelecekteki araştırmalar sorumlu bir geliştirme sağlamak için yönetişimi ele almalıdır. Ortak öncelikler arasında, ajanın araç seçiminde adaleti sağlamak, potansiyel veri kümesi sızıntılarına karşı savunma ve en önemlisi, otomatikleştirilmiş karar iş akışları için net hesap verebilirlik zincirlerini sürdürmek yer alır. MCP'nin teknik yeteneklerine uygun yönetişim protokolleri oluşturmak, protokolün uzun vadeli sürdürülebilir büyümesini ve sorumlu uygulamasını sağlamak için çok önemlidir.[2]

Alıntılanan çalışmalar

1. Making REST APIs Agent-Ready: From OpenAPI to MCP ... - arXiv, erişim tarihi

Ekim 13, 2025, https://arxiv.org/abs/2507.16044
2. Model Bağlam Protokolü (MCP): Manzara, Güvenlik Tehditleri ... - arXiv, erişim tarihi Ekim 13, 2025, https://arxiv.org/pdf/2503.23278
3. Model Bağlam Protokolü (MCP) Nedir | Nasıl Çalışır - Kodexo Labs, erişim tarihi Ekim 13, 2025, https://kodexolabs.com/what-is-model-context-protocol-mcp/
4. NBER WORKING PAPER SERIES AI AGENTS FOR ECONOMIC ..., erişim tarihi Ekim 13, 2025, https://www.nber.org/system/files/working_papers/w34202/w34202.pdf
5. LiveMCP-101: MCP özellikli sistemlerin stres testi ve tanısı... - arXiv, erişim tarihi Ekim 13, 2025, https://arxiv.org/abs/2508.15760
6. MCPToolBench++: A Large Scale AI Agent Model Context ... - arXiv, erişim tarihi Ekim 13, 2025, https://arxiv.org/abs/2508.07575